Trabajo Fin de Grado
Grado en Ingeniería Aeroespacial

Desarrollo de una aplicación para autocorrección de ejercicios de modelado de piezas industriales 3D

Autor: Jaime Areñas Morales

Tutoras: Laura García Ruesgas | Cristina Torrecillas Lozano

Dpto. Ingeniería Gráfica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado
Grado en Ingeniería Aeroespacial

Desarrollo de una aplicación para autocorrección de ejercicios de modelado de piezas industriales 3D

Autor:
Jaime Areñas Morales

Tutoras:
Dña. Laura García Ruesgas | Dña. Cristina Torrecillas Lozano

Dpto. Ingeniería Gráfica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado: Desarrollo de una aplicación para autocorrección de ejercicios de modelado de piezas industriales 3D

Autor: Jaime Areñas Morales
Tutoras: Dña. Laura García Ruesgas | Dña. Cristina Torrecillas Lozano

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Quisiera aprovechar estas líneas para agradecer a las personas que han contribuido directa o indirectamente a desarrollar el presente proyecto, que marca el culmen a mi etapa universitaria.

A mis tutoras Dña. Laura García Ruesgas y Dña. Cristina Torrecillas Lozano por el soporte prestado durante la realización del trabajo. Su total disposición, su cercanía en las tutorías, y las facilidades prestadas han sido claves para que el proyecto llegue a buen puerto.

A mis amigos, por las risas y los buenos momentos vividos durante tantos años.

A mi familia, por depositar siempre su confianza en mí. A mis abuelos, por el apoyo incondicional que me han dado durante todos estos años. Y a mi hermana, por ser mi inspiración.

Por último, mi más profundo agradecimiento a mis padres. Por vuestro cariño, por vuestra paciencia infinita y por el sacrificio que habéis hecho para formarme académicamente y como persona. Haber culminado esta etapa es un logro tanto mío como vuestro.

Jaime Areñas Morales
Sevilla, 2022

Resumen

En el campo de la docencia, existen una serie de procedimientos automatizados para autoevaluar ejercicios o exámenes de múltiples materias. Algunos como el empleo de test de respuesta única o el uso de problemas de solución cerrada donde solo se valora la respuesta final del alumno, son métodos contrastados comúnmente empleados. La complejidad aumenta a la hora de autoevaluar un texto, pudiendo optar por la búsqueda de las palabras clave para asignar una puntuación automática. En el ámbito de la Ingeniería Gráfica, en los entornos CAD/CAM, la evaluación emplea la resolución de ejercicios que tiene como salida dibujos en papel o modelos digitales 2D o 3D, a los que dar una valoración automatizada no es una tarea sencilla. El contenido del presente proyecto intenta ofrecer una posible vía para la autocorrección automatizada de ejercicios de modelado de piezas industriales 3D. La idea del proyecto no es generar un programa que corrija los fallos de un modelo 3D automáticamente, sino crear una aplicación que fomente la interacción con el usuario. Se busca que este entienda dónde ha errado y sepa cómo solventarlo mediante la información que el programa le facilita.

Para lograrlo, se estudia los posibles campos de aplicación del lenguaje de macros de *Microsoft*, *Visual Basic for Applications (VBA)*, al programa de diseño informático asistido por ordenador CATIA V5R19. Se ha creado un programa, al que se le ha llamado CPIT (Corrector de Piezas Industriales Tridimensionales), capaz de comparar un modelo 3D con un archivo de referencia. La metodología empleada se basa en extraer parámetros generales de la pieza, como el volumen o el centro de gravedad, las operaciones que la componen, y las dimensiones 2D y 3D para realizar comparaciones entre ambos modelos, en concreto compara aquellas operaciones que sean comunes en la pieza modelada y en la referencia. Al final del proceso de análisis, estima una puntuación acorde a las discrepancias encontradas, y aconseja al usuario qué hacer para mejorar el modelo realizado.

Se llevó a cabo un perído de prueba del programa desarrollado, en el que se analizaron piezas de distinta índole con el fin de mostrar la versatilidad que ofrece el software. Adicionalmente, se comprobó que las puntuaciones de las piezas evaluadas fueran acordes a las discrepancias encontradas durante el proceso de análisis. Se concluyó que el programa desarrollado analizaba correctamente los modelos y establecía la puntuación acorde a los resultados obtenidos.

El software desarrollado cumple con los objetivos marcados al comienzo del proyecto, destacando por ser una herramienta fácil de usar, por la capacidad que ofrece para autoevaluar piezas de distinta índole, y por ser un programa instructivo para los usuarios ya que le proporciona una serie de consejos para mejorar sus modelos. Consecuentemente, es un software útil en la docencia, ofreciendo una posible vía para la corrección automatizada de exámenes.

Abstract

In the field of teaching, there are a number of automated procedures for self-assessment of exercises or exams in multiple subjects. Some methods, such as the use of single answer tests or the use of closed solution problems where only the student's final answer is evaluated, are commonly used. The complexity increases when it comes to self-assessing a text, with the option of searching for keywords to assign an automatic score. In the field of Graphic Engineering, in CAD/CAM environments, the evaluation uses the resolution of exercises that result in paper drawings or 2D or 3D digital models, to which giving an automated assessment is not an easy task. The content of this project attempts to offer a possible way for automated self-correction of 3D industrial parts modeling exercises. The idea of the project was not to generate a program that would correct the failures of a 3D model automatically, but to create an application that would encourage interaction with the user. The idea is to understand where the user has made a mistake and to know how to fix it through the information provided by the program.

To achieve this, the possible areas of application of the Microsoft macro language, Visual Basic for Applications (VBA), to the computer-aided computer design software CATIA V5R19 are studied. A program has been created, CPIT (Three-dimensional Industrial Parts Corrector), which is capable of comparing a 3D model with a reference file. The methodology used is based on extracting general parameters of the part, such as volume or center of gravity, the operations that compose it, and the 2D and 3D values to make comparisons between both models, specifically comparing those operations that are common in the modelled part and in the reference. At the end of the analysis process, it estimates a score according to the discrepancies found, and advises the user what to do to improve the model.

A test period of the developed program was carried out, in which parts of different kinds were tested in order to show the versatility offered by the software. In addition, it was verified that the scores of the pieces evaluated were in accordance with the discrepancies found during the analysis process. It was concluded that the developed program correctly analyzed the models and established the score according to the results obtained.

The developed software fulfills the objectives set at the beginning of the project, standing out for being an easy-to-use tool, for the ability it offers to self-evaluate parts of different kinds, and for being an instructive program for the users because it provides a number of tips to improve the models. Consequently, it is a useful software in education, offering a possible way of automated correction of exams.

Índice

Índice de Figuras

Índice de Tablas

Índice de Códigos

1 Introducción

CATIA V5 (*Computer-Aided Three dimensional Interactive Application*) es un programa informático de diseño y fabricación empleado en múltiples sectores como el aeronaútico, la automoción o la construcción. Al ser una herramienta potente para el modelado sólido de piezas industriales, suele impartirse en las titulaciones relacionadas con estos sectores, como el caso del Grado en Ingeniería Aeroespacial.

El aprendizaje del diseño de piezas no es algo mecánico, basado en la simple explicación de lo que hace cada herramienta, requiere de la comprensión de ejemplos prácticos. La evaluación de este aprendizaje suele realizarse mediante el contraste con piezas definidas con el modelado "más óptimo". Sin embargo, no existen comparadores de piezas, como los ya existentes comparadores de documentos de texto, en el entorno CAD/CAM, siendo el principal objetivo de este trabajo.

1.1 Introducción al problema

La educación siempre ha estado en continua evolución, desde la forma de impartir las clases hasta las distintas alternativas de realizar y corregir un examen. Por lo general, esta última es la actividad en la que un docente invierte más tiempo, por lo que se ha desarrollado una clara tendencia a emplear múltiples métodos de autocorrección de ejercicios. Con la pandemia que nos ha tocado vivir estos últimos años todos estos cambios se han acelerado, motivados por las distintas alternativas que los docentes han buscado para evitar que el alumno copie durante sus pruebas. De esta forma, se ha optado comunmente por el uso de aplicaciones que autoevalúen los exámenes, dando un tiempo límite para la realización de los mismos y empleando herramientas, como *webcam* y micrófonos, para que todo salga según lo previsto.

Cuando se piensa en autoevaluar una actividad se imagina el típico test con una o con múltiples respuestas predefinidas. Sin embargo, las técnicas de autocorrección son muy dispares, desde algunas puramente numéricas hasta otras donde el texto juega un papel primordial.

Desde el punto de vista numérico, destacan técnicas de autoevaluación como valorar solo el resultado de un problema, sin tener en cuenta la metodología empleada. En el caso de tener que autocorregir un texto, se puede optar por la búsqueda de palabras clave. Sin embargo, la interpretación del contenido es una dificultad añadida en este tipo de ejercicios.

Esta práctica se complica cuando el ejercicio a evaluar no es numérico ni alfanumérico. Asignaturas

relacionadas con la Ingeniería Gráfica, entornos CAD/CAM, no emplean textos o números, usándose comunmente dibujos o archivos modelo para corregir el resultado obtenido por el alumno. El problema se complica al corregir mecánicamente ejercicios de modelado 3D. [1].

Trabajar en esta línea es la base de partida del presente proyecto, en el que se exponen una serie de parámetros a tener en cuenta a la hora de analizar y puntuar una pieza en base a una referencia dada.

1.2 Entorno CAD/CAM con CATIA

El Diseño y la Fabricación Asistidos por Ordenador (CAD/CAM) es una disciplina que estudia el diseño y la fabricación de cualquier tipo de producto mediante el empleo de sistemas informáticos como herramienta de soporte. El grado de desarrollo de los sistemas CAD/CAM ha supuesto una auténtica revolución que afecta no solo al dibujo y al diseño de ingeniería, sino que condiciona y transforma los procesos de fabricación y los constructivos, modificando su metodología operacional. El término CAD (*Computer-Aided Design*) se puede definir como el uso del ordenador para ayudar en el diseño de una parte individual, un subsistema o un sistema completo. El proceso de diseño puede estar en el nivel de concepto del sistema o en el nivel de diseño detallado de una parte. Por otro lado, el término CAM (*Computer-Aided Manufacturing*) se puede definir como el uso de sistemas informáticos para la planificación, gestión y control de las operaciones de una planta de fabricación mediante una interfaz directa o indirecta entre el sistema informático y los recursos de producción. [2] y [3].

Centrándonos en la asignatura de Diseño y Fabricación Asistidos por Ordenador, impartida en el Grado en Ingeniería Aeroespacial entre otros, el programa informático empleado para introducir los entornos CAD/CAM es CATIA V5, que es uno de los software de diseño líder en todo el mundo para el modelado de productos. Fue desarrollado por *Dassault Systèmes* y, aunque inicialmente fue creado para servir en la industria aeronáutica, se emplea en una gran variedad de sectores como la construcción o la automoción. CATIA ofrece la posibilidad de desarrollar cualquier cosa por muy compleja que sea, destacando la precisión con la que trabaja y su sencillez de uso.

Provee una arquitectura abierta para el desarrollo de aplicaciones o para personalizar el programa. Las interfaces de programación de aplicaciones, CAA2 (o CAAV5), se pueden programar en Visual Basic y C++. [4].

CATIA está compuesto de una serie de módulos que ofrecen distintas herramientas. Este proyecto se ha centrado en el módulo *Part Design*, que permite al usuario crear sólidos 3D, destacando por la libertad de uso y el grado de complejidad que pueden alcanzar los modelos creados. Del gran cátalogo de operaciones que presenta este módulo, se han estudiado aquellas que se consideran básicas y más usadas a la hora de crear piezas, siendo un conjunto de dieciséis operaciones que permiten generar una gran cantidad de modelos de distinta índole. Si se desea conocer este grupo de operaciones, puede consultar el *Capítulo 3*, concretamente la sección 3.2.8.

Figura 1.1 Módulo *Part Design* de CATIA.

Debido a la gran variedad de posibilidades que ofrece CATIA, la autoevaluación de los modelos 3D realizados en este software es compleja. Por ejemplo, el análisis del árbol de modelado (conjunto ordenado de operaciones que el usuario emplea para realizar una pieza) en muchos casos resulta de poca ayuda, ya que se pueden optar por distintos caminos para llegar a un mismo resultado. Por otro lado, corregir manualmente cada una de las operaciones que conforman el modelo realizado supone una tarea laboriosa, debido a la gran variedad de parámetros con las que se pueden definir. [1]. Estas son algunas de las razones por las que se ha decidido poner inicio a este proyecto.

1.3 Introducción al entorno de programación VBA en CATIA V5

VBA (*Visual Basic for Applications*) es el lenguaje de macros de Visual Basic v6, que ha sido incorporado en multitud de aplicaciones para aumentar su funcionalidad. CATIA en 1998 con la versión V5 incorporó VBA a su entorno, pudiendo realizar macros en VB y en lenguaje C++, siendo aún los lenguajes de macros que se ha dispuesto para su versión V6. A pesar de ser una gran alternativa para el desarrollo de aplicaciones, presenta desventajas como la necesidad de disponer del entorno donde se desarrolla una macro para poder compilarla, o la poca versatilidad para funcionar en otros sistemas operativos. [5].

Para acceder a *Visual Basic* en CATIA, se debe seleccionar **Tools - Macro - Visual Basic Editor**. Para poder abrir este editor es necesario instalarlo previamente, ya que su instalación es independiente a la de CATIA. [6].

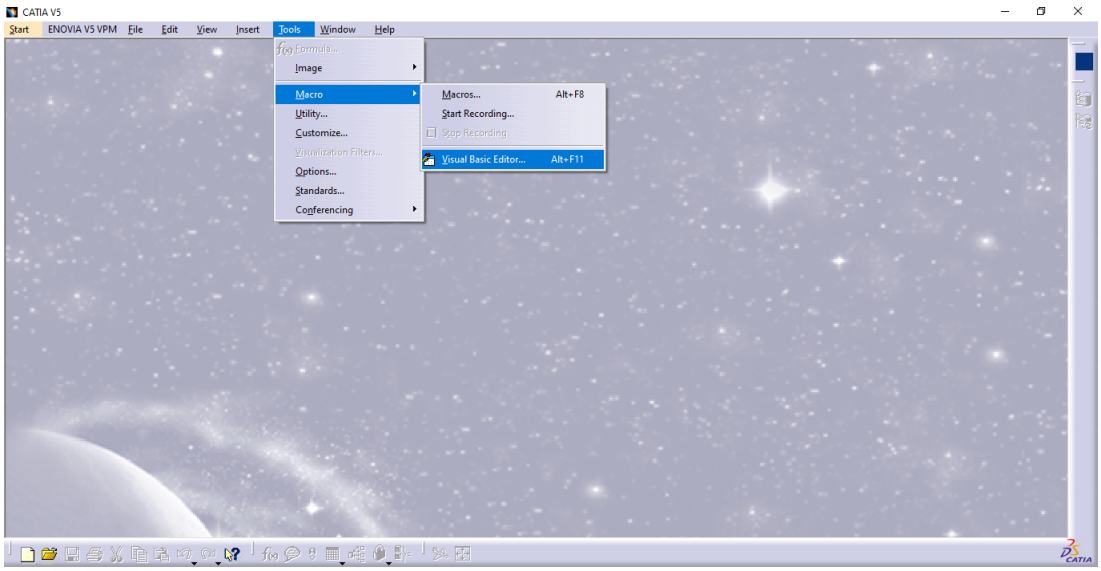


Figura 1.2 Acceso a *Visual Basic Editor*.

Antes de entrar en detalle con VBA, es conveniente explicar qué es una macro. Consiste en un conjunto de funciones escritas en un lenguaje de programación que agrupa una serie de comandos. Estos permiten realizar automáticamente las operaciones requeridas, lo que supone un ahorro en tiempo y la reducción de fallos humanos al realizar las mismas.

Las macros permiten una gran versatilidad a la hora de automatizar un proceso de diseño. Las posibilidades son prácticamente ilimitadas, como exportar valores de la pieza a una hoja de Excel o

crear elementos de forma automática entre otras.

Una vez entendido el concepto de las macros, se pasa a explicar el entorno de *Visual Basic Editor*.

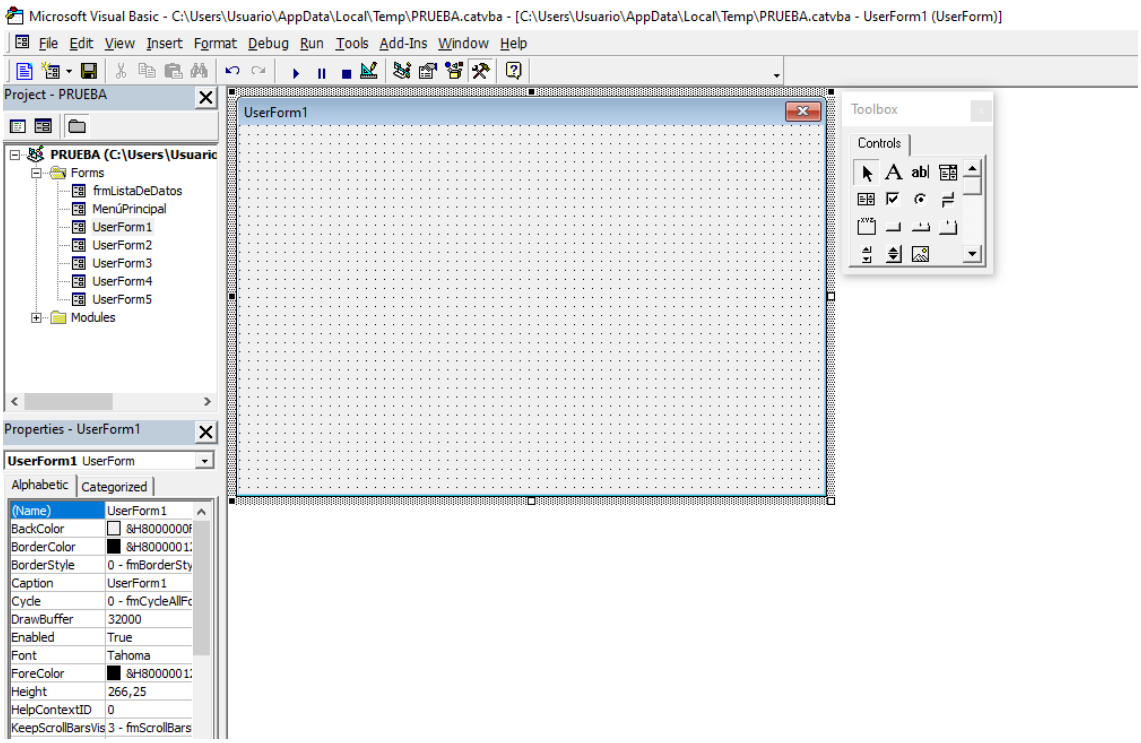


Figura 1.3 Interfaz *Visual Basic Editor*.

La *Figura 1.3* representa la ventana que aparece al abrir el editor, en la que se pueden ver varias partes claramente diferenciadas.

A la izquierda se encuentran los menús *Project Explorer* y *Properties Window*, que se pueden activar entrando en la pestaña *View* del programador. [5].

- ***Project Explorer***: recoge todos los archivos del proyecto de trabajo (*Project*). Estos se clasifican en formularios (*UserForm*), módulos (*Module*) y clases (*Class Module*). Para introducirlos en el proyecto activo, se debe entrar en la pestaña *Insert* y clicar sobre el deseado. Se exponen los dos primeros, que son los empleados en este proyecto:
  - *UserForm*: son ventanas gráficas en las que, mediante la caja de herramientas (*Toolbox*), se añaden botones, texto, imágenes, etc. Permiten iteracción entre el usuario y el programa, y se guardan con extensión .frm.
  - *Module*: son rutinas independientes que pueden ser llamadas desde distintos formularios del programa. Por tanto, suelen almacenar funciones que pueden ser empleadas por cualquier formulario. Se almacenan con extensión .bas.
- ***Properties Window***: recoge en forma de columna todas las propiedades del archivo activo en la ventana *Project Explorer*.

En la parte superior de la *Figura 1.3* aparece el directorio del archivo en el que se está trabajando y las distintas ventanas que nos ofrece el editor. Cabe destacar la ventana *Help*, que nos permite acceder a *Microsoft Visual Basic Help*. Ver *Figura 1.4*.

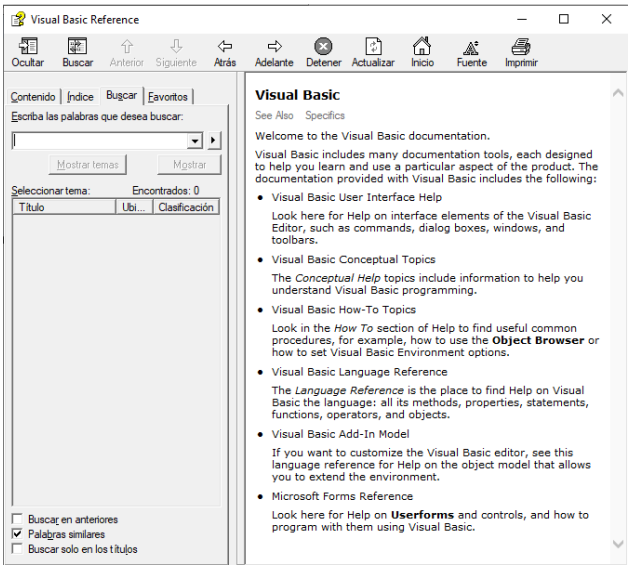


Figura 1.4 Ayuda en *Visual Basic Editor*.

En esta ventana se pueden buscar las dudas que se tengan sobre el funcionamiento de los comandos de VBA, mostrándose en la derecha explicaciones sobre cómo usarlos y algunos ejemplos de código.

Por último, destacar la ventana *Object Browser*, a la que se accede desde *View*. Esta dispone de una barra de búsqueda en la que se puede introducir el nombre de un objeto que no se entienda cómo funciona, dándonos como resultado tres columnas de información que nos ayudarán a manejar el mismo. La primera columna hace referencia a la librería, la segunda al tipo de objeto y la última a las operaciones que se pueden realizar. Además, en la parte inferior de la ventana aparece una descripción de cómo deben ser las entradas de la operación para que funcione correctamente.

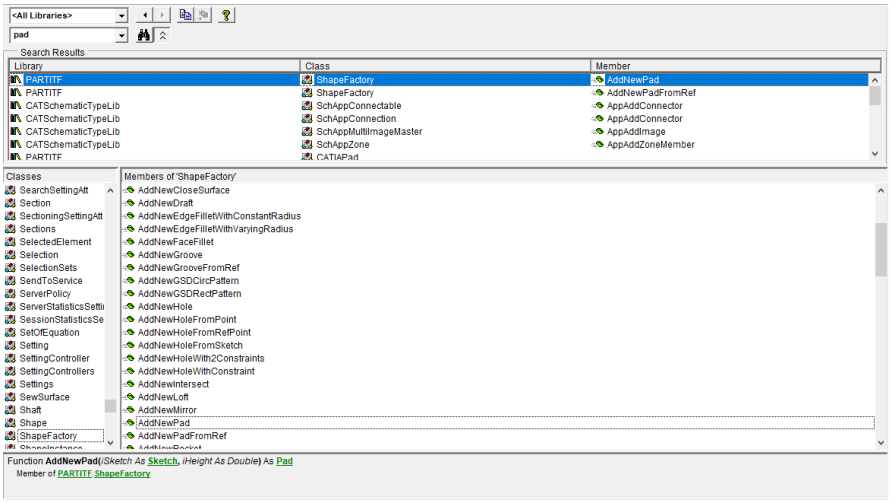


Figura 1.5 *Object Browser* VBA.

1.4 Motivación y objetivos del proyecto

En el ámbito del aprendizaje del diseño de piezas industriales 3D, sería de gran ayuda disponer de una aplicación que comparase una pieza modelada por el usuario con otra de referencia. Desde un punto de vista docente, ayudaría para la preparación de un examen, ya que el alumno podría poner a prueba sus conocimientos realizando las piezas modeladas por el profesor en clase y viendo si estas coinciden entre sí. Aunque la libertad en el diseño de una pieza es difícil de evaluar, existen ciertos parámetros físicos y necesidades de diseño propias de algunas herramientas como, por ejemplo, los agujeros roscados, que son imprescindibles en la pieza. En base a la existencia o no de ciertas operaciones y a la cercanía en esos parámetros físicos, puede establecerse una nota y es una buena base de partida para que el profesor sepa si el resultado se adecua a los requerimientos establecidos.

La solución que se expone en esta memoria es una aplicación creada mediante CATIA V5R19 VBA, que compara la pieza realizada por el usuario con una pieza o una plantilla (extensiones *.CATPart* y *.xlsx* respectivamente) de referencia. El algoritmo generado crea un documento Excel con tres hojas en las que se muestran los parámetros generales de ambas piezas, los resultados de analizar cada una de las operaciones que las conforman, y las discrepancias existentes. Además, se ha fomentado la interacción entre el programa y el usuario, es decir, este último tiene capacidad de escoger el tipo de análisis a realizar, y el programa le va mostrando por la ventana de resultados del menú principal el estado del análisis: errores encontrados, consejos para mejorar la pieza, etc. No se pretende que el software desarrollado, al que se le ha denominado CPIT (Corrector de Piezas Industriales Tridimensionales), corrija los fallos del modelo automáticamente, lo que se busca es que el usuario entienda dónde ha errado y sepa cómo solventarlo.

Los objetivos que se buscan con el programa son:

- Ofrecer al alumno una vía de autocorrección de piezas que le prepare mejor de cara a la realización del examen práctico.
- Programa interactivo, indicando los posibles problemas de diseño para la corrección de los mismos y la re-evaluación de la pieza.
- Autocorrección de piezas basándonos en una referencia y a varios niveles de detalle, incluyendo:
  - Mostrar los parámetros generales de la pieza y hallar las discrepancias con los de la de referencia.
  - Analizar el árbol de modelado de la pieza. Mostrar las operaciones que se han empleado y el número que hay de cada tipo.
  - Analizar cada una de las operaciones para buscar dónde residen los errores. Esto se lleva a cabo solo con aquellas operaciones que aparezcan tanto en el árbol de modelado de la pieza como en el de la de referencia.
  - Analizar el sketch de cada operación, que igualmente se lleva a cabo solo con las operaciones que pertenecen a ambas piezas.

1.5 Estructura del proyecto

El proyecto se ha desglosado en los siguientes capítulos:

- **Capítulo 2**: se exponen las distintas piezas que se usan como referencia para comprobar la validez del programa realizado. Son tres piezas de distinta dificultad para asegurar el correcto funcionamiento de la aplicación. Además, se adjuntan los planos de modelado de cada una de ellas.

- **Capítulo 3**: comienza con una introducción a la programación en VBA. Posteriormente, se detallan los módulos programados en VBA y cómo interactúan con el usuario y entre sí. Se introducen los objetivos de cada módulo, las entradas que necesitan y los códigos que cada uno emplea para lograr su cometido.

- **Capítulo 4**: se comparan distintas piezas con las referencias presentadas previamente en el primer capítulo, y se muestran los resultados obtenidos por la aplicación.

- **Capítulo 5**: se exponen las conclusiones realizadas al finalizar el proyecto. Se evalúa la línea de trabajo seguida, y se determina si los objetivos marcados al comienzo del trabajo se han cumplido.

- **Capítulo 6**: se aportan una serie de ideas y mejoras para realizar en el futuro, con el fin de completar el proyecto.

2 Establecimiento de parámetros y piezas de estudio

Una de las aplicaciones más frecuentes de CATIA es el modelado sólido de piezas industriales. Para ello, una de las herramientas que ofrece este software es el *Part Design*, en la que el usuario va definiendo una serie de operaciones que en su conjunto conforman el modelo. En esta sección se introducen los parámetros que el software desarrollado evalúa y las piezas que posteriormente se analizarán en el *Capítulo 4*. El objetivo es ver que la aplicación es capaz de corregir cualquier pieza realizada en CATIA V5R19, introduciendo los distintos niveles de análisis que emplea.

2.1 Parámetros a controlar

La aplicación se centra en una serie de parámetros que se han considerado suficientes para analizar su validez. El análisis de la pieza se divide en tres niveles de detalle:

- **Nivel 1**: Analiza los parámetros generales de la pieza. CPIT evalúa diez parámetros que aparecen en la herramienta *Measure Inertia* de CATIA V5R19, a la que se puede acceder desde *Toolbars - Measure*.
- **Nivel 2**: Analiza el árbol de modelado y el número de operaciones de la pieza. Adicionalmente, se lleva a cabo un análisis tridimensional de aquellas operaciones que forman parte tanto del árbol de modelado de la pieza como de la referencia.
- **Nivel 3**: Analiza los *sketches* de cada una de las operaciones. Al igual que ocurre en el análisis tridimensional, solo se analizan los de las operaciones que formen parte de ambas piezas.

Se ha desglosado el análisis para fomentar el aprendizaje del usuario. La idea es que a medida que se vaya ejecutando el programa le informe a través de la ventana de resultados sobre el estado del análisis y las correcciones que debe realizar. Esta metodología obliga al usuario a saber dónde ha cometido sus errores y cómo solucionarlos. Además, el hecho de separar el programa en distintos niveles permite distribuir la puntuación de la pieza entre multitud de aspectos. Consecuentemente, la nota final es lo más completa y precisa posible. Por otro lado, remarcar que los niveles van en orden creciente de detalle, desde los parámetros generales de la pieza (Nivel 1) hasta el análisis bidimensional de las operaciones que la conforman (Nivel 3).

En la *Tabla 2.1* se recogen los parámetros que se van a considerar.

Tabla 2.1 Parámetros que CPIT evalúa.

Tipo	Parámetro	Metodología de corrección	Técnica de verificación
Global	Posicionamiento en sistema de referencia	Obtención de coordenadas del *Bounding Box* de la pieza y de la referencia	Sustracción entre coordenadas de modelos. Correcto:0; Incorrecto: $\neq 0$
Global	Volumen	Obtención del volumen computado en la aplicación	Sustracción entre cifras de modelos. Correcto:0; Incorrecto: $\neq 0$
Global	Centro de masas	Consultar coordenadas del centro de gravedad	Ídem anterior
Global	Momentos de inercia	Consultar valores numéricos	Ídem anterior
Global	Orden de operaciones	Comprobación entre árboles de modelado	Correcto: Tienen el mismo número de operaciones Incorrecto: No tienen el mismo número de operaciones
Local	Dimensiones 3D de las operaciones	Consultar valores numéricos	Sustracción entre cifras de modelos. Correcto:0; Incorrecto: $\neq 0$
Local	Dimensiones 2D de las operaciones	Consultar valores numéricos	Ídem anterior

2.2 Piezas modelo

Para comprobar la validez de la aplicación se van a analizar una serie de piezas, que se han escogido en orden creciente de aprendizaje. De esta forma se prueba la capacidad de CPIT para evaluar piezas de distinta índole. El objetivo de esta sección es introducirlas, exponiendo las operaciones que las conforman y su nivel de dificultad. Estas serán las referencias que posteriormente se usarán para comparar otras piezas.

Aunque CPIT es capaz de evaluar un modelo realizado por el usuario con una plantilla en la que se almacenen los datos de referencia (archivo *.xlsx*), se ha optado por exponer en esta memoria solo comparaciones con archivos *.CATPart*, ya que se clarifican las discrepancias existentes entre los modelos analizados. Para disipar cualquier duda existente, en el *Capítulo 3* se explica cómo es capaz el programa de analizar la pieza del usuario mediante un archivo de referencia de extensión *.xlsx*.

Antes de introducir las piezas, cabe destacar que los planos de modelado adjuntados han sido facilitados por el departamento de Ingeniería Gráfica de la ETSI (Escuela Técnica Superior de Ingeniería), Universidad de Sevilla. Concretamente, estos planos pertenecen a la asignatura de Diseño y Fabricación Asistidos por Ordenador (DAFO), impartida en el tercer curso del Grado en Ingeniería Aeroespacial.

2.2.1 Pieza 1: pieza introductoria

Es una de las primeras piezas empleadas en la asignatura para dar a conocer CATIA a los alumnos. Las operaciones a las que se tiene que enfrentar CPIT son las principales del menú *Sketch-Based Features*:

- *Pad*: es una de las operaciones más empleadas en CATIA, por lo que es fundamental que se analice correctamente. El objetivo de CPIT es obtener la altura, tanto si el *pad* es *mirrored extent* como si no.
- *Pocket*: operación comúnmente usada. El reto que se le presenta a CPIT con esta operación es identificar la profundidad.
- *Hole*: es una de las operaciones con más parámetros tridimensionales. Será importante identificar el tipo de *hole* que el usuario emplea, ya que los valores a analizar dependen de este parámetro. Además, será crucial identificar correctamente el punto de referencia a partir del que se lleva a cabo esta operación.
- *Edge Fillet*: los redondeos son empleados con asiduidad en CATIA. Los retos que se le presentan a CPIT son analizar el radio y la longitud de las esquinas que se redondean.

Las operaciones que conforman esta pieza son:

Tabla 2.2 Operaciones de la Pieza 1.

Tipo de operación	Número de unidades
Pad	1
Pocket	1
Hole	2
Edge Fillet	1

Su modelado realizado en CATIA V5R19 queda:

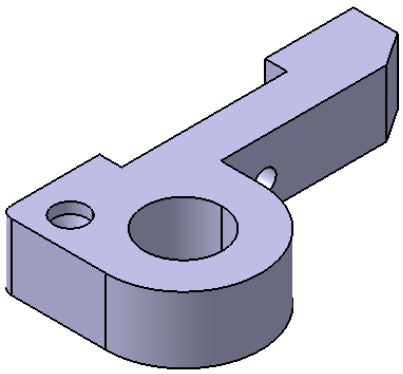


Figura 2.1 Pieza 1 en CATIA.

Esta pieza, debido a su sencillez, será de ayuda para comprobar si CPIT funciona correctamente. Es interesante ya que está compuesta por las operaciones más empleadas en CATIA, por lo que se podrá ver si la aplicación evalúa correctamente los parámetros de las mismas.

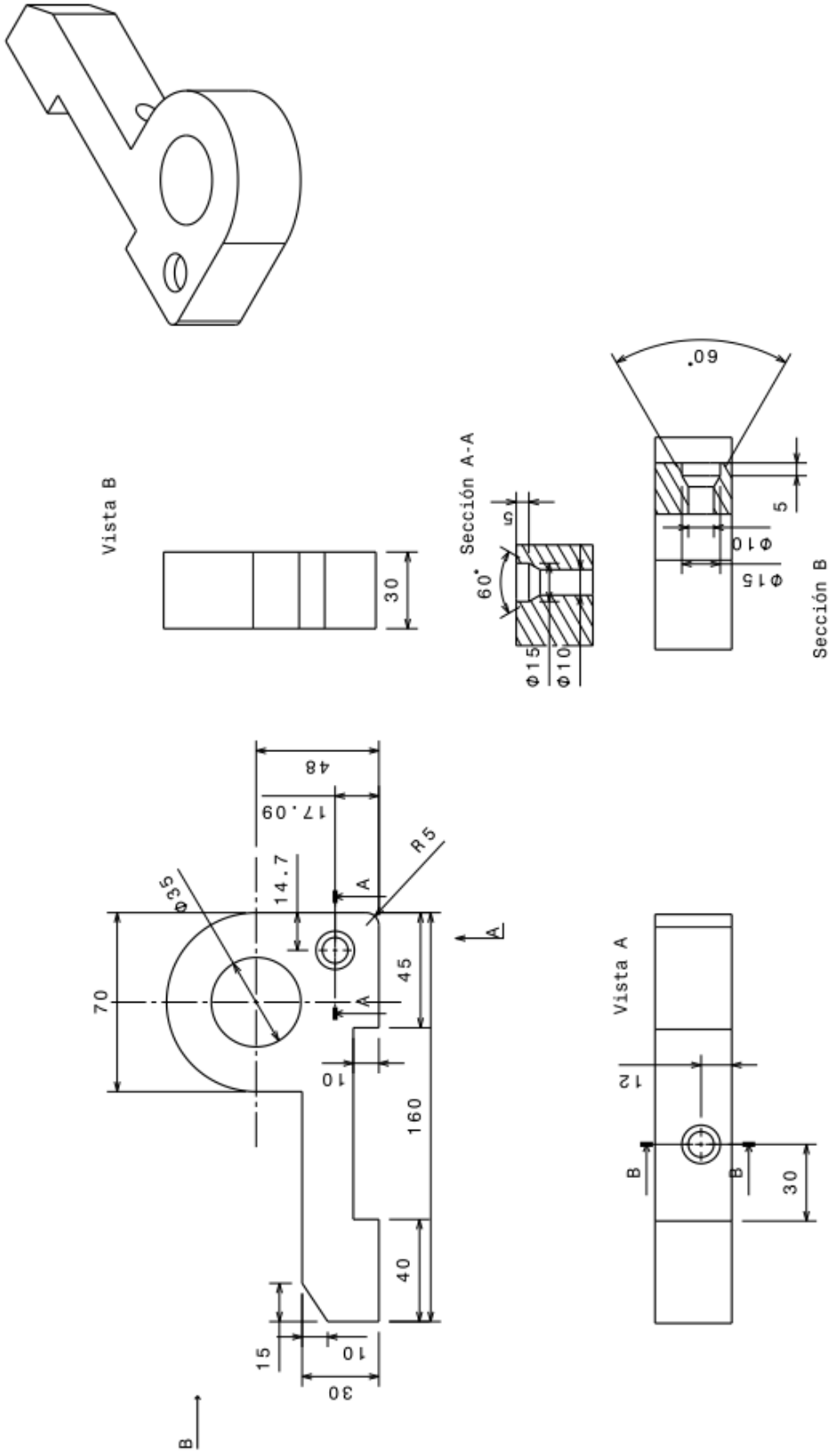


Figura 2.2 Plano de modelado de la pieza 1.

2.2.2 Pieza 2: codo circular

Se trata de un codo circular con base rectangular, modelado en la asignatura de Diseño y Fabricación Asistidos por Ordenador del Grado en Ingeniería Aeroespacial. Si se compara con la primera pieza analizada, aparecen nuevas operaciones a las que se tendrá que enfrentar la aplicación:

- *Rib*: es una operación un tanto especial, ya que está definida por dos *sketches* y no tiene parámetros tridimensionales, al igual que el *slot*. CPIT ha de ser capaz de identificar los *sketches* independientemente de los tres casos en los que se pueden configurar:
  1. Ambos definidos dentro de la operación.
  2. Uno definido dentro de la operación y otro en el árbol de modelado.
  3. Ambos definidos en el árbol de modelado, fuera de la operación.
- *Circular Pattern*: los patrones, tanto rectangular como circular, son comumente empleados en CATIA. Los retos que se le presentan a CPIT con esta operación son:
  - Identificar el objeto que se copia y el número de veces que se repite.
- *Stiffener*: los refuerzos son de gran utilidad en el modelado de las piezas. CPIT ha de ser capaz de identificar el espesor y el número de refuerzos que se realizan.

Las operaciones que conforman esta pieza son:

Tabla 2.3 Operaciones de la Pieza 2.

Tipo de operación	Número de unidades
Pad	3
Pocket	1
Hole	1
Edge Fillet	4
Rib	1
Circular Pattern	3
Stiffener	1

Su modelado en CATIA V5R19 queda:

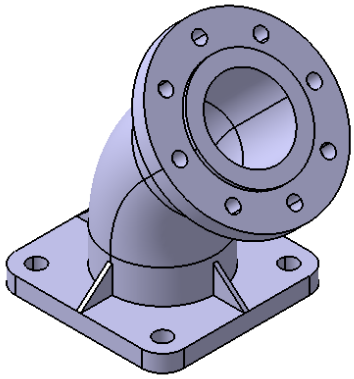


Figura 2.3 Pieza 2 en CATIA.

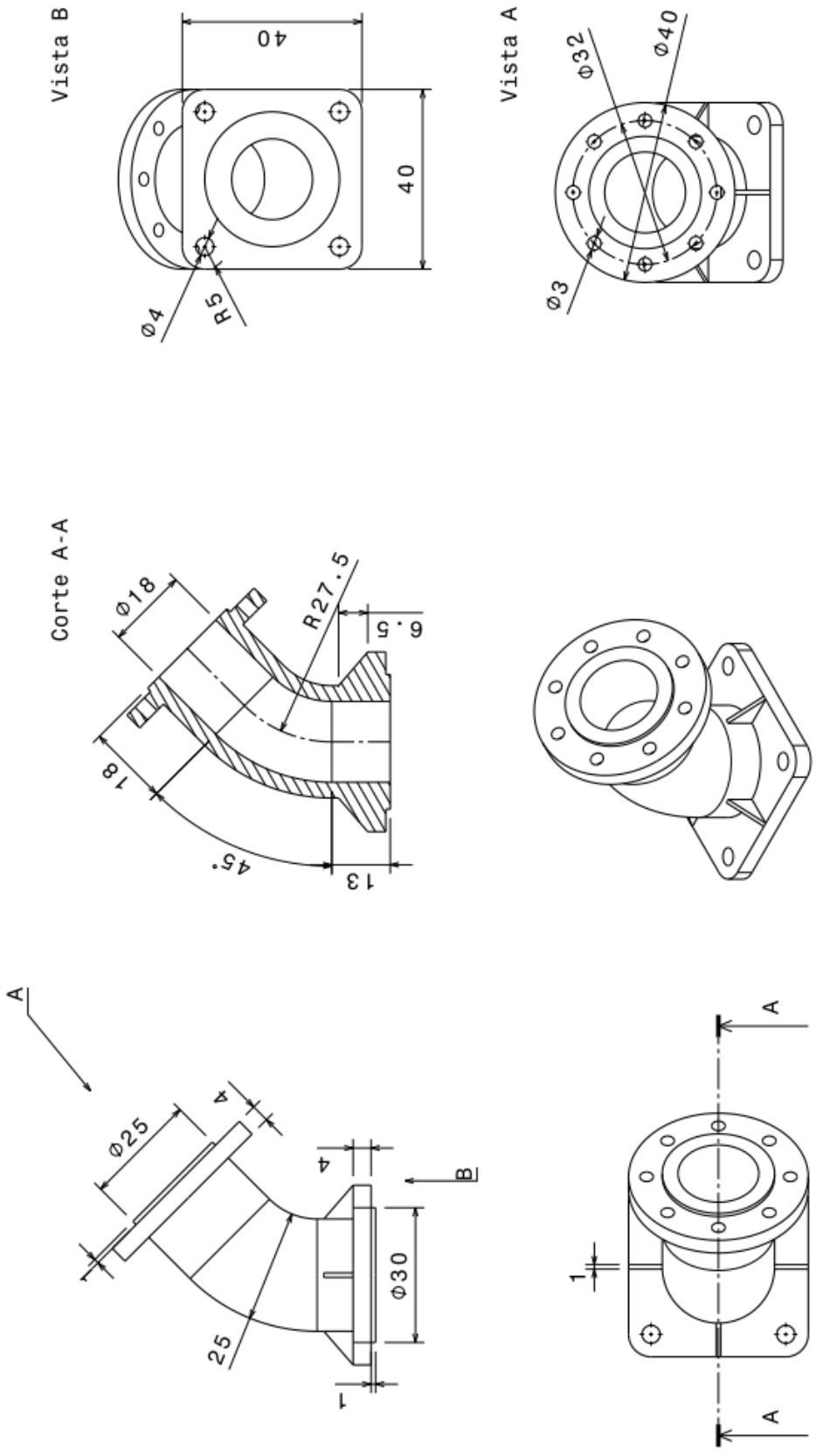


Figura 2.4 Plano de modelado de la pieza 2.

2.2.3 Pieza 3: asiento con respaldo

Al igual que las piezas anteriores, también es una pieza modelada en la asignatura de DAFO. Este asiento con respaldo presenta nuevas operaciones a las que se tendrá que enfrentar CPIT:

- *Chamfer*: el chaflán es una operación que solo posee parámetros tridimensionales, es decir, no está definido por ningún *sketch*. Además, al igual que otras operaciones analizadas anteriormente como el *edge fillet*, se puede seleccionar más de un elemento al que realizar el mismo chaflán. Por tanto, los objetivos de CPIT han de ser:
  1. Identificar la longitud y el ángulo.
  2. Identificar el número de elementos a los que se le realiza el chaflán simultáneamente.
- *Rectangular Pattern*: al igual que en la *Pieza 2*, se puede encontrar un patrón en el árbol de modelado, aunque en esta ocasión es rectangular. Los retos que se le presentan a CPIT con esta operación son:
  - Identificar el objeto que se copia y el número de veces que se repite.
- *Mirror*: la simetría es una operación que simplifica el modelado de las piezas, ya que permite copiar un objeto respecto a un plano de referencia, evitando tener que realizarlo de nuevo. CPIT debe ser capaz de:
  1. Identificar el objeto al que se le aplica simetría.
  2. Identificar el plano de referencia empleado.

Las operaciones que la conforman son:

Tabla 2.4 Operaciones de la Pieza 3.

Tipo de operación	Número de unidades
Pad	4
Edge Fillet	1
Rib	4
Rectangular Pattern	2
Chamfer	2
Mirror	1

Su modelado en CATIA V5R19 queda:

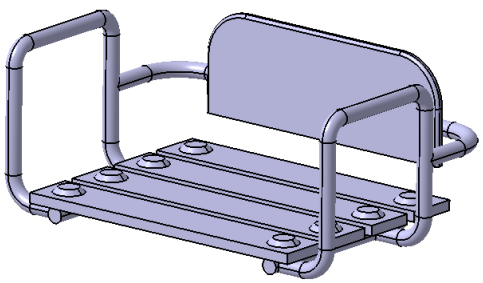


Figura 2.5 Pieza 3 en CATIA.

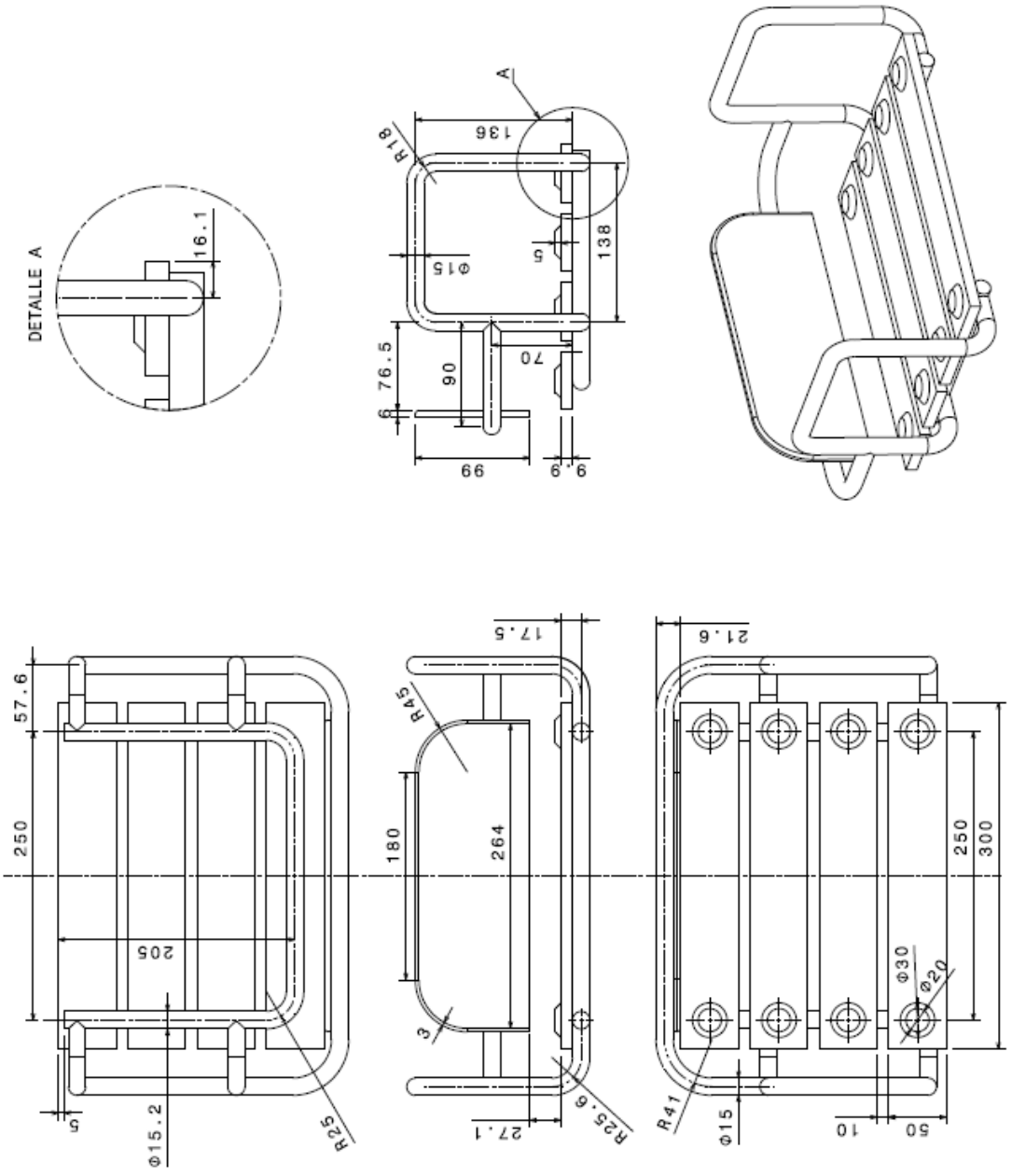


Figura 2.6 Plano de modelado de la pieza 3.

3 Metodología

En este capítulo se hace una introducción a la programación en VBA y se describen las macros empleadas en el desarrollo de la aplicación. Se presentan los distintos módulos empleados y la interacción entre ellos.

3.1 Cómo empezar a programar en VBA

Este apartado va a explicar cómo un usuario sin conocimientos en programación puede iniciarse en el entorno de *Visual Basic*.

Es importante saber que VBA es un lenguaje de programación por eventos, es decir, tanto la estructura como la ejecución de los programas van determinados por los sucesos que ocurran en el sistema, definidos por el usuario o que ellos mismos provoquen. Por tanto, se debe prestar especial atención a que los códigos empleados respondan adecuadamente a los distintos eventos a los que se dirige la aplicación. [5].

1). Declaración de estamentos

El primer paso a realizar antes de trabajar con cualquier parámetro es nombrarlo. De esta forma, se obtiene un primer acercamiento con el estamento que posteriormente se definirá.

Para definir el tipo de variable de un parámetro se emplea el comando *Dim*, que tiene la siguiente estructura:

```
    Dim Nombre As Tipo
```

Código 3.1 Ejemplo de declaración de una variable.

Existen multitud de tipos de variables, como:

- *String*: Empleada para declarar textos.
- *Double*: Empleada para declarar números.
- *Document*: Empleada para declarar documentos.
- *Integer*: Empleada para declarar *arrays*.

Por ejemplo, si se quiere declarar un texto se emplea el siguiente código:

```
Dim texto1 As String
```

Código 3.2 Declarar una variable string.

En el caso de que no se defina el tipo de variable, VBA identifica el parámetro como *variant*, es decir, lo declara como cualquier tipo de variable. El tipo *variant* no suele ser comunmente empleado, ya que supone un mayor tiempo de ejecución del código y un aumento de errores.

1.1). Estamentos

Son una instrucción completa que puede contener *keywords* (*And*, *if*, *for*, *while*, *sub*, *function*...), operadores ($+$, $-$, $*$, $/$, ...), variables, constantes y expresiones.

```
Set partDocument1 = CATIA.ActiveDoument
```

Código 3.3 Ejemplo de un estamento.

1.2). Estamentos Ejecutables

Se trata de acciones iniciales.

```
Select1.Search("name='Optimization.MinimumValue',all")
```

Código 3.4 Ejemplo de una subfunción.

2). Funciones y subfunciones

Son una secuencia de estados que conforman la operación deseada. Esta operación viene especificada en una función.

```
Sub mySubwithParameter (myParameter)
    MsgBox myParameter
End Sub
```

Código 3.5 Ejemplo de una subfunción.

La diferencia entre una función y una subfunción es que mientras la primera devuelve un valor la segunda no.

3). Estructuras condicionales e iterativas

- **Condicionales**: Visto que durante la programación muchas veces se llega a situaciones en que es conveniente utilizar condicionales para comprobar, por ejemplo, que un valor se encuentra en un intervalo aceptable, es conveniente hacer mención de la ejecución: *if...then...*

```
If [condición] Then
    [Estamento]
ElseIf [condición] Then
    [Estamento de ElseIf]
End If
```

Código 3.6 Estuctura del comando If...Then...End If.

- **Iterativas**: Otra de las estructuras usadas frecuentemente en la programación son aquellas en las que se realiza una misma operación varias veces consecutivas con el fin de encontrar un valor justo o que haga funcionar un programa hasta que cierta condición se cumpla.

Existen varias configuraciones, algunas de las cuales se comentan a continuación.

- For...Next

```
Contador
For [Contador] = [inicio] To [end] {paso a paso}
    [Estamentos]
Next
```

Código 3.7 Bucle For...Next.

- While...Wend

```
Contador
While [{Contador}Condición]
    [Estamentos]
Wend
```

Código 3.8 Bucle While...Wend.

- Do...Loop

```
Do [{While/Until}condition]
    [Estamentos]
    [Exit Do]
    [Estamentos]
    Loop
    Do [Estamentos]
[Exit Do]
[Estamentos]
Loop [{While/Until}condition]
```

Código 3.9 Bucle Do...Loop.

4). Objetos orientados a la programación

Los objetos son una parte de memoria en la que está contenida cierta información y metodología que permite operar con dicha información almacenada. Requiere de un cierto código especial para

hacer que funcione, ya que dispone de comportamientos definidos por métodos específicos tales como:

- Pasar información a través de parámetros.
- Operaciones de cálculo que pueden:
  - Cambiar cierta parte de los datos iniciales.
  - Diseñar ciertas operaciones que no están por defecto.
  - Devolver valores de los datos iniciales.
  - Devolver resultados de los cálculos usando tanto la información introducida externamente como de la contenida en el objeto.

5). Cómo definir un objeto

Para cada objeto en particular se define una clase (*classe*) que sirve como plantilla en la que se recoge cómo operan los objetos y los datos que contienen. Es conveniente resaltar que una clase puede ser utilizada para hacer funcionar uno o más objetos.

CLASE	OBJETO
Describe la estructura del objeto	Es el resultado de la clase
Es una plantilla	Tiene una copia única de toda variable no-estática pero no de las variables tipo clase (*static*)
Especifica la representación de la información, el comportamiento, la interrelación (vía variables, métodos y *parents*-estructura lógica)	

Figura 3.1 Diferencia entre objeto y clase.

6). Interacción CATIA VBA - usuario

Se va a explicar cómo crear una interacción entre VBA y el usuario, mediante la creación de una interfaz con la herramienta *UserForm*. Para crear una nueva se debe acceder a **Insert-UserForm**.

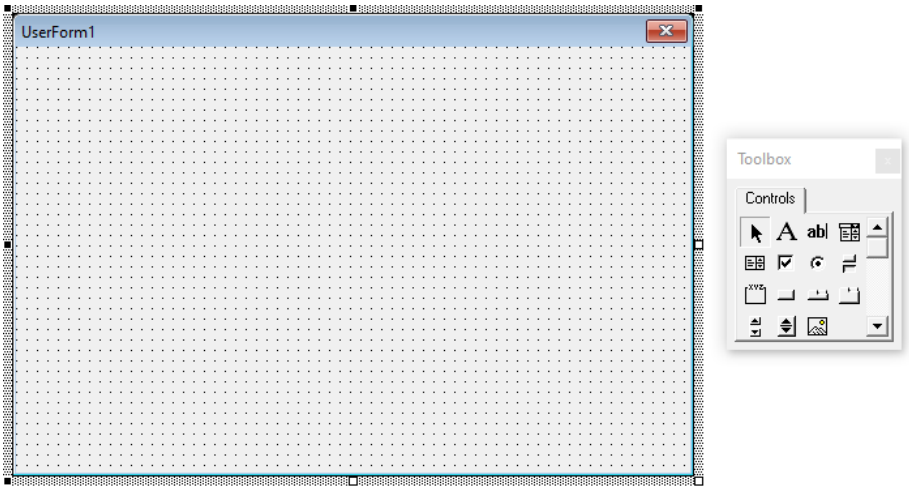


Figura 3.2 *UserForm* en VBA.

Como se puede observar en la imagen adjunta, a la derecha del panel del *UserForm* aparece una ventana con el nombre *Toolbox*. Mediante la misma, el usuario puede modelar la interfaz a su gusto con la introducción de una serie de elementos.

Símbolo	Etiqueta	Descripción
	*Label*	Permite escribir títulos o comentarios.
	*TextBox*	Permite al usuario introducir texto.
	*ListBox*	Control en el que se muestran varios registros, pudiendo seleccionar uno o más de uno.
	*ComboBox*	Control parecido al ListBox con una propiedad llamada *Style*, que permite 3 formas distintas de presentar una lista.
	*CheckBox*	Permite seleccionar una opción al usuario.
	*OptionButton*	Permite seleccionar una opción al usuario.
	*ToggleButton*	Botón para selección de opciones.
	*Frame*	Agrupar diferentes objetos referidos a un mismo tema.
	*CommandButton*	Permite ejecutar un evento.
	TabStrip	Separadores o etiquetas.
	*MultiPage*	Contenedor para una colección de objetos.

Figura 3.3 Elementos disponibles en *Toolbox*.

Al introducir cualquiera de ellos, si el usuario clica dos veces sobre el mismo, le aparecerá el editor del *Userform*. Dentro del mismo, se podrán introducir diferentes códigos en función de lo que se vaya buscando. Por ejemplo, si se desea establecer un botón que al clicarlo se cierre el programa, se tiene que añadir un *CommandButton* y asignarle el siguiente código:

```
Private Sub CommandButton2_Click()
    Unload Me
End Sub
```

Código 3.10 Salir del programa.

Por otro lado, *Visual Basic* dispone de los llamados *modules*, los cuales se pueden llamar desde un *UserForm*. A la hora de diseñar un programa, se puede crear una interfaz que vaya llamando distintos módulos que contienen el código que permite hacer distintas funciones. Por ejemplo, para llamar a una subfunción, se puede emplear el siguiente código:

```
Call Principal.Principal(directorio,ListBox1,ListBox2,OptionButton1,
    OptionButton2)
```

Código 3.11 Entradas del módulo Principal.

El comando *Call* llama a la subfunción *Principal*, que tiene una serie de entradas que corresponden a los diferentes elementos añadidos en el *UserForm*.

3.2 Diseño de programa

El objetivo de esta sección es mostrar cómo está CPIT estructurado, presentándose los distintos módulos empleados y la función de cada uno de ellos. En primer lugar, se va a introducir la metodología empleada por el programa para llevar a cabo el análisis de una pieza.

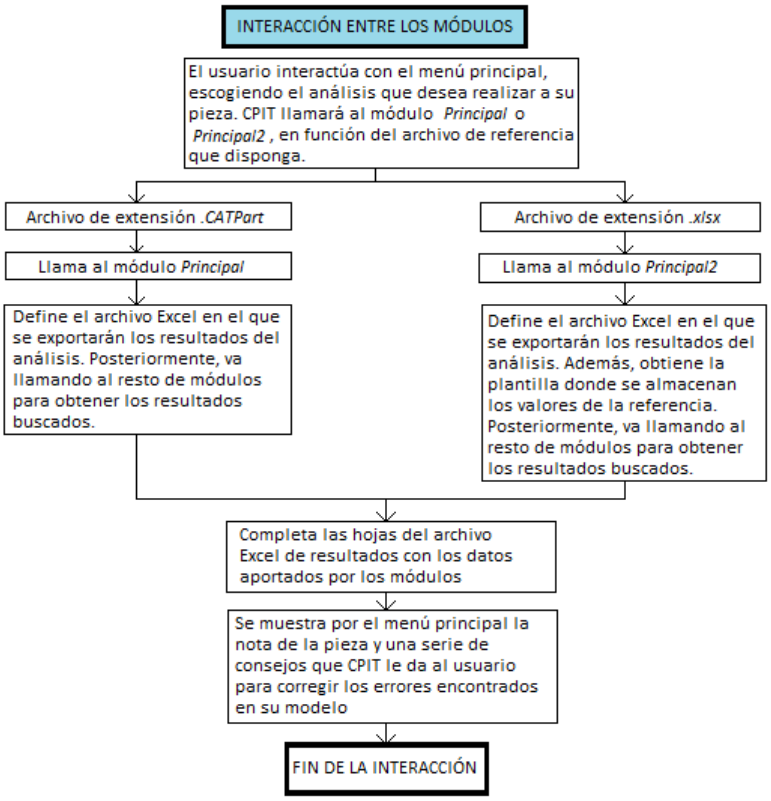


Figura 3.4 Diagrama sobre la metodología general de análisis empleada por CPIT.

Lo primero que tiene que realizar el usuario es descargar la macro del programa y añadirla a la librería de macros de CATIA, consultar *Anexo. Manual de usuario*. Hecho esto, debe acceder a los módulos del programa y ejecutar el denominado *Abrir_CPIT*, cuya función es abrir el menú del software desarrollado.

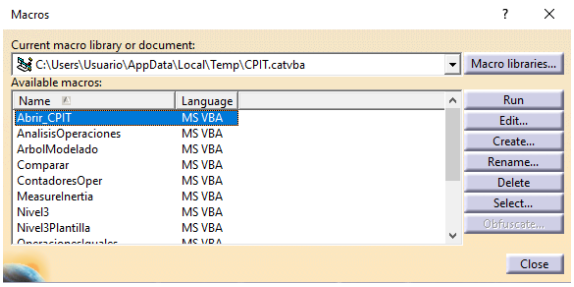


Figura 3.5 Ejecutar macro.

El código del módulo *Abrir_CPIT* tiene dos funciones: abrir el menú del programa y copiar el

directorio de la pieza que se va a analizar. De esta forma, cuando se abra el menú aparecerá en la parte superior el nombre de la pieza que está abierta en CATIA, pretendiendo que el usuario sepa en todo momento el modelo que va a analizar el software desarrollado. Además, si se intenta ejecutar la macro sin tener una pieza abierta en CATIA, saltará un mensaje advirtiendo que para que se ejecute el programa debe haber una pieza abierta.

```
Dim documento As Document
If CATIA.Windows.Count = 0 Then
    MsgBox "No se ha podido ejecutar el programa porque no hay
        ninguna pieza abierta", vbExclamation, "Error"
ElseIf CATIA.Windows.Count <> 0 Then
    Set documento = CATIA.ActiveDocument
    Menu.TextBox3 = documento.FullName
    Menu.Show
End If
```

Código 3.12 Código del módulo *Abrir_CPIT*.

Otro módulo fundamental para el correcto funcionamiento de la macro es el denominado *MeasureInertia*, que se encarga de aplicar sobre la pieza la herramienta *Measure Inertia* de CATIA. En esta aparecen los parámetros generales del modelo, como el volumen o el centro de gravedad, que son valores que el software desarrollado tiene en cuenta a la hora de autoevaluar la pieza modelada. Si la pieza ya tiene aplicada esta herramienta antes de ejecutar la macro, el módulo actualiza las medidas inerciales para tener en cuenta cualquier modificación que se haya realizado en el modelo.

```
Dim selBody, selMeasure As Selection
Set selBody = CATIA.ActiveDocument.Selection
Set selMeasure = CATIA.ActiveDocument.Selection
selMeasure.Search "Name=Area,all"
If selMeasure.Count = 0 Then
    selBody.Search "'Part Design'.Body,all"
    CATIA.StartCommand ("Measure Inertia")
    Dim WshShell As Object
    Set WshShell = CreateObject("wscript.shell")
    WshShell.AppActivate "CATIA V5"
    CATIA.RefreshDisplay = True
    WshShell.AppActivate "c:Measure Inertia"
    WshShell.SendKeys "{TAB 6}", True
    WshShell.SendKeys "{ENTER}", True
    Set WshShell = Nothing
ElseIf selMeasure.Count <> 0 Then
    selMeasure.Search "Name=Measure,all"
    CATIA.StartCommand ("Local Update")
    selMeasure.Clear
End If
```

Código 3.13 Código del módulo *MeasureInertia*.

A continuación, se presentan todos los módulos empleados por CPIT.

Tabla 3.1 Módulos que emplea CPIT.

Tipo	Nombre	Función
UserForm	MenuPrincipal	Interactúa con el usuario. Permite escoger distintos archivos de referencia y distintos tipos de análisis.
Module	Abrir_CPIT	Abre el menú principal. Es el módulo que se ejecuta desde la librería de macros.
Module	MeasureInertia	Aplica la herramienta *Measure Inertia* sobre la pieza, que es fundamental para extraer los parámetros generales del modelo.
Module	Principal	Va llamando al resto de módulos e informa al usuario a través de la ventana de resultados. Se activa cuando existe pieza de referencia.
Module	Principal2	Misma función que el módulo Principal. Sin embargo, se activa cuando no existe pieza de referencia.
Module	ParametrosGenerales	Extrae los parámetros generales de la pieza y de la referencia, y los compara.
Module	ArbolModelado	Extrae el árbol de modelado de la pieza y de la referencia, y comprueba si tienen el mismo orden y el mismo número de operaciones.
Module	AnalisisOperaciones	Extrae las dimensiones 3D de cada operación de la pieza y de la referencia. Solo se analizan aquellas operaciones que se encuentren en el árbol de modelado tanto de la pieza como de la referencia.
Module	PintarResultadosAnalisisOper	Exporta los resultados del módulo *AnalisisOperaciones* al archivo Excel de resultados.
Module	ContadoresOper	Extrae el número de operaciones de cada tipo que hay tanto en la pieza como en la referencia.
Module	OperacionesIguales	Extrae el tipo de las operaciones que tienen en común la pieza y la referencia.
Module	Nivel3	Extrae las dimensiones 2D de las operaciones de la pieza y de la referencia, cuando esta última se encuentra en un archivo de extensión *.CATPart*. Solo se analizan aquellas operaciones que se encuentren en el árbol de modelado tanto de la pieza como de la referencia.
Module	Nivel3Plantilla	Extrae las dimensiones 2D de las operaciones de la referencia, en el caso de que esta se encuentre en un archivo de extensión *.xlsx*. Solo se analizan aquellas operaciones que se encuentren en el árbol de modelado tanto de la pieza como de la referencia.
Module	Comparar	Comprueba si los valores 3D y 2D de las operaciones de la pieza y de la referencia coinciden. Además, informa al usuario a través de la ventana de resultados sobre las correcciones que debe realizar y establece una puntuación en base a los errores encontrados.

El funcionamiento de la aplicación es secuencial. Primero el usuario interactua con el menú principal escogiendo el análisis deseado. Posteriormente, una vez se inicie el programa, se activa el módulo Principal o Principal2 dependiendo de la referencia disponible. Por último, este irá llamando al resto de módulos para ir ejecutando los distintos niveles de análisis e irá informando al usuario a través de la ventana de resultados.

A continuación se van a explicar en detalle los módulos presentados previamente.

3.2.1 MenúPrincipal

Le permite al usuario interactuar con la aplicación, eligiendo entre distintos tipos de análisis. La guía de uso del menú viene recogida en el *Anexo. Manual de usuario*.

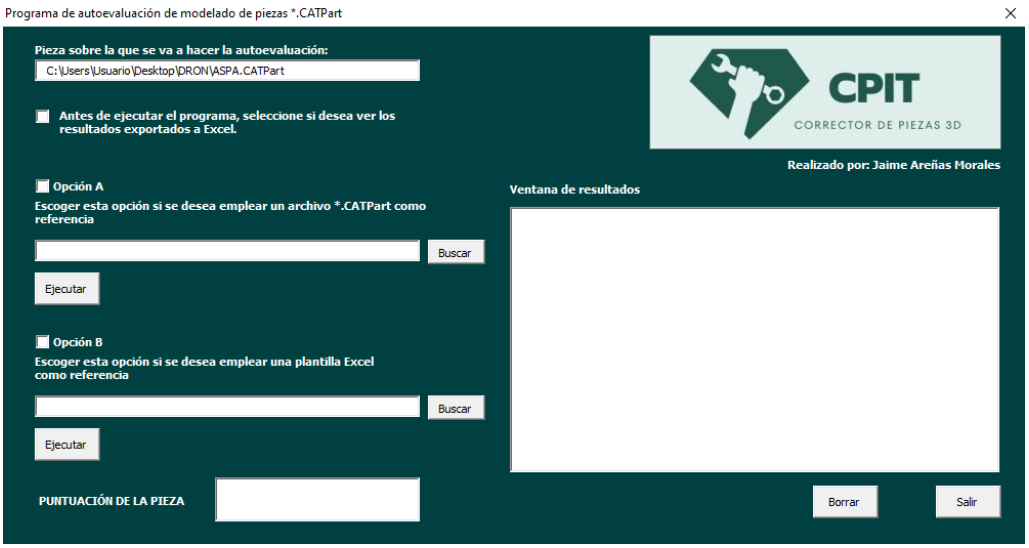


Figura 3.6 Menú principal de CPIT.

La ventana del menú principal está compuesta por:

- **Pieza a analizar**: en el cuadro de texto superior del menú, se muestra el directorio de la pieza que está abierta en CATIA antes de ejecutar la macro del programa. De esta forma, se pretende que el usuario se asegure de que la pieza que CPIT va analizar coincida con la que él desea.

- **Mostrar los resultados**: el menú principal da la posibilidad al usuario de escoger si desea ver los resultados obtenidos. Dado que CPIT informa a través de la ventana de resultados sobre el análisis y correciones posibles a realizar, no es necesario tener que mostrar las soluciones que se van obteniendo. Si se marca la casilla superior del menú, se mostrará el archivo Excel de soluciones una vez finalice el proceso de análisis. Independientemente de la opción escogida, los resultados quedan guardados en la misma carpeta que el archivo de referencia empleado.

- **Opción A y Opción B**: asociadas a emplear una pieza o una plantilla Excel como referencia, respectivamente. Una vez se activa una de las opciones, la otra queda inhabilitada. Además, al marcar cualquier opción, se aplica sobre la pieza modelada por el usuario la herramienta

Measure Inertia de CATIA.

- **Buscar**: al clicar sobre este botón se abre una ventana que permite al usuario seleccionar el archivo de referencia. Una vez abierto, se copia el directorio del mismo en el cuadro de texto adyacente. La selección del archivo de referencia se lleva a cabo con el comando *CATIA.FileSelectionBox*, que tiene tres entradas: nombre de la ventana emergente, tipo de archivo (.CATPart o .xlsx, en nuestro caso) y modo de selección (0 para abrir y 1 para guardar). En el caso de escoger la opción A, la pieza de referencia se abre en CATIA y se le aplica la herramienta *Measure Inertia* al igual que al modelo del usuario.

- **Ejecutar**: su función es ejecutar el software, es decir, llamar a los módulos Principal y Principal2 aportándoles una serie de entradas. Ver *Tabla 3.2*.

Tabla 3.2 Entradas de los módulos Principal y Principal2.

Nombre	Función
Directorio	Almacena el directorio de la referencia, ya sea el de la pieza o el del archivo Excel. Al pulsar los botones de búsqueda del menú principal y seleccionar el archivo de referencia, su directorio queda guardado en el cuadro de texto adyacente. Con el comando *TextBox1.text* se lee y se guarda en una variable *string*.
ListBox1	Es la ventana de resultados. A medida que la aplicación se vaya ejecutando, irá informando a través de ella sobre el estado del análisis y los posibles fallos de la pieza.
ListBox2	Es la ventana de puntuación de la pieza. A medida que la aplicación se vaya ejecutando, la nota se actualiza en función de los errores que se vayan encontrando.
Checkbox1	Si está activado, la aplicación muestra el archivo Excel de soluciones. Si está desactivado, no se muestra.

- **Ventana de resultados**: indica si los niveles de análisis están ejecutándose con éxito, y muestra las discrepancias existentes entre la pieza y la referencia.

- **Ventana de puntuación de la pieza**: una vez ejecutado el programa, CPIT muestra la nota asociada a la pieza, que se ha calculado a partir de los errores que tiene.

- **Borrar y Salir**: como sus nombres indican, al clicar sobre estos botones se borran los datos introducidos en el menú y se sale del programa, respectivamente. Para limpiar el menú, CPIT desactiva el botón de opción de análisis (*Checkbox.Value = False*), y limpia la ventana de resultados (*ListBox1.Clear*), el cuadro de puntuación (*ListBox2.Clear*), y el cuadro del directorio (*TextBox=""*). Para salir del programa, se usa el comando *Unload Me*.

Es primordial que el usuario conozca perfectamente cómo interactuar con el menú principal, ya que es la base de partida que usa CPIT para saber qué debe analizar. Por esta razón, se ha realizado un diagrama de flujo para ahondar en esta idea.

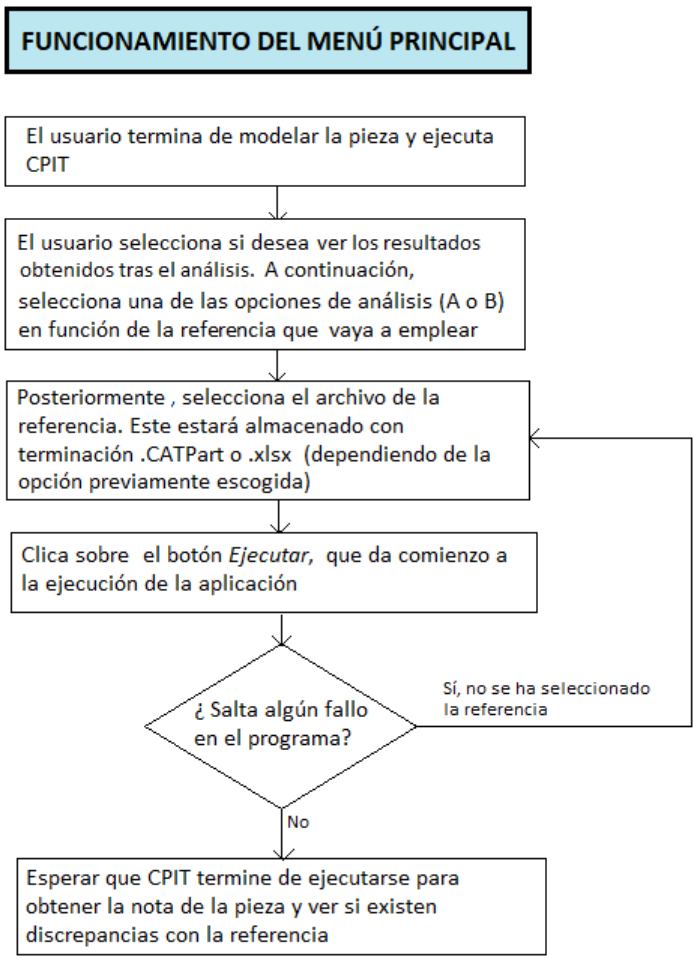


Figura 3.7 Diagrama de flujo sobre el funcionamiento del menú principal.

3.2.2 Principal

Módulo principal al que CPIT accede si se dispone de pieza de referencia. Es una subfunción (*Sub*), por lo que no devuelve ningún valor. Tiene una serie de tareas asignadas:

- Generar el archivo de Excel en el que aparecerán los resultados de las mediciones realizadas.
- Definir las ventanas de CATIA.
- Definir el *product* y el *part* de la pieza y de la referencia.
- Llamar al resto de módulos. Estos son en su mayoría funciones, por lo que devuelven valores que serán determinantes en el desarrollo del análisis.
- Informar al usuario sobre cómo avanza el programa.
- Finalizar la ejecución de la aplicación en caso de que el usuario no interactúe correctamente con el menú principal. Además, le informa del fallo que ha cometido.
- Cerrar el archivo *.CATPart* de la referencia una vez finalice el análisis.

Este módulo recibe una serie de entradas que provienen de la interacción del usuario con el menú principal de CPIT. Estas son:

- El directorio donde se encuentra almacenada la referencia.

- *ListBox1*, que es la ventana de resultados del menú principal.
- *ListBox2*, que es la ventana en la que se muestra la nota que recibe la pieza del usuario.
- *Checkbox1*.

En primer lugar, este módulo define una variable denominada *indicador* que toma el valor cero para indicar que se tiene pieza de referencia. En el caso del submódulo Principal2, el indicador valdrá uno para señalar que en lugar de disponer de una pieza en CATIA, los valores de la referencia se almacenan en una plantilla Excel. Posteriormente, pasa a definir el directorio de la referencia y a analizar las entradas que recibe para ver si la interacción entre el usuario y el menú ha sido correcta.

El directorio es una variable *string* que se obtiene como entrada del módulo. En cuanto a las entradas recibidas, pueden darse los siguientes casos:

- Si el directorio no está vacío se define el archivo Excel de soluciones. Este estará compuesto de tres hojas que recogerán cada uno de los niveles de análisis. Para su creación se emplea el siguiente código:

```
Set objexcel = CreateObject("Excel.Application")
Set objWorkbook = objexcel.workbooks.Add()
objWorkbook.Sheets.Add Count:=2
Set objsheet1 = objWorkbook.Sheets.Item(1)
Set objsheet2 = objWorkbook.Sheets.Item(2)
Set objsheet3 = objWorkbook.Sheets.Item(3)
objsheet1.name = "NIVEL 1"
objsheet2.name = "NIVEL 2"
objsheet3.name = "NIVEL 3"
```

Código 3.14 Definición del archivo Excel de resultados.

A continuación, se introducen una serie de comandos útiles para modificar las hojas previamente definidas. Con el comando *objsheet.Cells(,) = " "* se puede escribir en las celdas, introduciendo la fila y la columna entre paréntesis y lo que se quiera añadir entre apóstrofes. Por otro lado, con *objsheet.Range().Interior.color = RB(, ,)* se les puede asociar un color, escribiendo la celda en el primer paréntesis y el color en el último (dado por tres dígitos). También se pueden combinar varias celdas empleando *objsheet1.Range(" : ").Merge* y remarcar el borde con *ojsheetbjsheet1.Range(" : ").Borders.color = RGB(, ,)*, introduciendo el rango de celdas que se desean modificar.
- Si el directorio de la referencia está vacío, se avisa al usuario de que seleccione una referencia y se sale del programa.

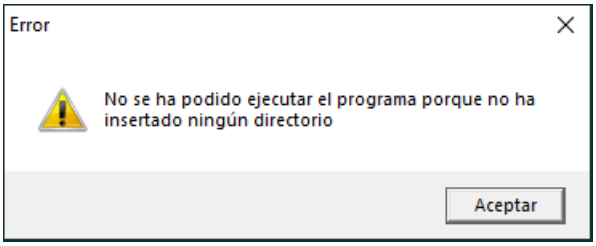


Figura 3.8 Error por no introducir el directorio del archivo de referencia.

Una vez se han analizado los posibles errores del usuario, se definen las ventanas de CATIA donde se encuentran tanto la pieza del usuario como la de la referencia.

```
Dim ventanaPieza As Window
Dim ventanaRef As Window
Set ventanaPieza = CATIA.Windows.Item(1)
Set ventanaRef = CATIA.Windows.Item(2)
```

Código 3.15 Definición de las ventanas de CATIA.

Ya abiertas ambas piezas, se definen el documento, el *product* y el *part* de cada una. Por ejemplo, para el caso de la pieza modelada por el usuario se emplea el siguiente código:

```
ventanaPieza.Activate
Dim documento As Document
Set documento = CATIA.ActiveDocument
Dim productDocument As Product
Set productDocument = documento.Product
Dim partDocument As Part
Set partDocument = documento.Part
```

Código 3.16 Definición del documento, part y product de la pieza.

Realizados estos pasos previos, el módulo procede a llamar a las funciones que realizarán los distintos niveles de análisis. Estas devolverán las notas que se le asocian a la pieza en cada nivel. El siguiente diagrama recoge el peso que tiene cada módulo en la nota final.

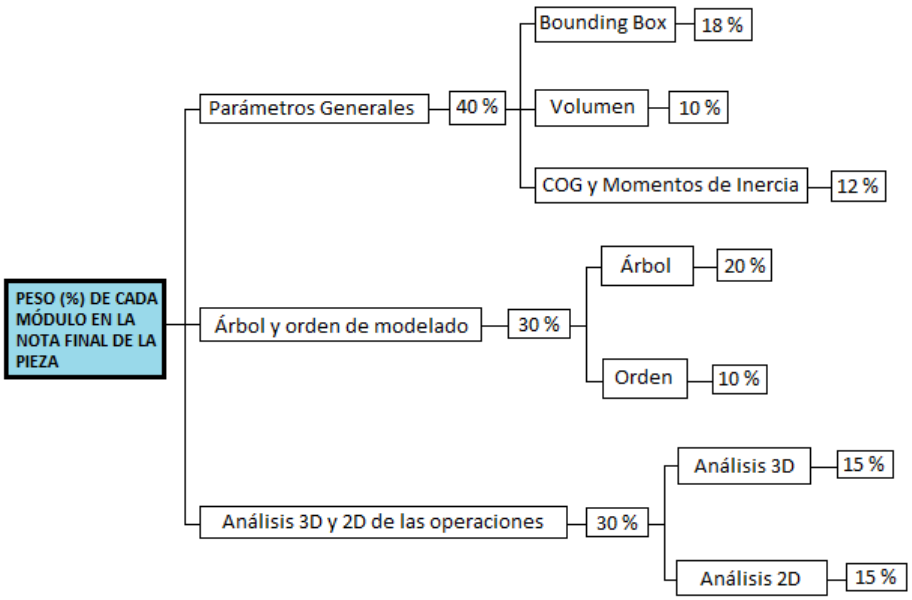


Figura 3.9 Peso de cada módulo en la puntuación de la pieza.

En primer lugar, llama a la función *ParametrosGenerales*, que devuelve en la variable *fun1* la nota que le corresponde a la pieza tras el nivel 1 de análisis. Si la función se ha ejecutado sin problema,

se le avisa al usuario a través de la ventana de resultados.

Si la variable *fun1* es distinta de cuatro, significa que se han encontrado discrepancias entre los valores de los parámetros generales de la pieza y de la referencia. En este caso, CPIT muestra a través de la ventana de resultados el siguiente mensaje:

- INCORRECTO: Existen diferencias entre los parámetros de tu pieza y de la referencia.

En caso contrario, si la variable *fun1* es igual a cuatro, significa que la pieza y la referencia son iguales en el nivel 1 de análisis. En este caso, CPIT muestra a través de la ventana de resultados el siguiente mensaje:

- CORRECTO: Los parámetros de tu pieza y de la referencia son iguales.

En segundo lugar, el módulo *Principal* llama a la función *ArbolModelado*, que devuelve en la variable *fun2* la nota asociada a la pieza tras analizar su árbol de modelado. Al igual que con la función anterior, se le informará al usuario sobre el estado del análisis. La metodología empleada por CPIT para puntuar el árbol y el orden de modelado se recoge en la sección 3.2.5.

Una vez obtenidas las salidas de ambas funciones (fun1 y fun2), se suman para obtener la puntuación de la pieza tras los primeros análisis. Este valor será una de las entradas del módulo *Comparar*, que es el encargado de mostrar por pantalla la nota final de la pieza. Antes de analizar las operaciones que conforman la pieza, se llama a las funciones *OperacionesIguales* y *ContadoresOper*, que se encargan de identificar las operaciones que aparecen tanto en la pieza como en la referencia, y de contar el número de ellas, respectivamente.

A continuación, se llama a la función *AnalisisOperaciones* tanto para la pieza como para la referencia. Esta devolverá una matriz con las operaciones que tienen en común ambas piezas, almacenando el nombre, el tipo y los parámetros tridimensionales característicos de estas. Posteriormente, se llama a la subfunción *PintarResultadosAnalisisOper* para exportar los resultados obtenidos a la hoja 2 del archivo Excel.

El programa sigue con la subfunción *Nivel3*, que examina los sketches de las operaciones y exporta los resultados a la hoja 3 del archivo Excel. Esta subfunción también es llamada tanto para la pieza como para la referencia. En el caso de disponer de una plantilla Excel de referencia, se llama a la subfunción *Nivel3Plantilla* para evaluar los parámetros bidimensionales de la referencia.

El último módulo llamado es *Comparar*, que se encarga de asignarle una nota a la pieza en base al análisis bidimensional y tridimensional. Por último, muestra en el menú principal la puntuación que CPIT estima oportuna para la pieza.

Para finalizar, si el usuario desea exportar los resultados se muestra el archivo Excel y se cierra la pieza de referencia. El código empleado es el siguiente:

```
If CheckBox1.Value = True Then
    objexcel.Visible = True
End If
documento2.Close
```

Código 3.17 Mostrar archivo Excel de resultados.

3.2.3 Principal2

Módulo principal al que CPIT accede si se dispone de una plantilla Excel de referencia, en lugar de un archivo *.CATPart*. Es una subfunción, por lo que no devuelve ningún valor. Tiene una serie de tareas asignadas:

- Generar el archivo de Excel en el que aparecerán los resultados de las mediciones realizadas.
- Definir la ventana de CATIA de la pieza del usuario.
- Definir el *product* y el *part* de la pieza del usuario.
- Llamar al resto de módulos. Estos son en su mayoría funciones, por lo que devuelven valores que serán determinantes en el desarrollo del análisis.
- Informar al usuario sobre cómo avanza el programa.
- Finalizar la ejecución de la aplicación en caso de que el usuario no interactúe correctamente con el menú principal. Además, le informa del fallo que ha cometido.
- Analizar la plantilla de referencia para comparar la información que recoge con los parámetros extraídos de la pieza del usuario.

En cuanto a las entradas que recibe esta subfunción, son las mismas que las del módulo *Principal*.

- El directorio donde se encuentra almacenada la referencia.
- *ListBox1*, que es la ventana de resultados del menú principal.
- *ListBox2*, que es la ventana en la que se muestra la nota que recibe la pieza del usuario.
- *Checkbox1*.

El funcionamiento de este módulo es prácticamente igual al de *Principal*, realizando ciertas modificaciones:

1. La variable *indicador* vale uno en lugar de cero, indicando que el archivo de la referencia tiene extensión *.xlsx*.
2. Definición de la plantilla Excel de referencia y de sus hojas. Para ello, se ha empleado el código:

```
Set objexcelPlantilla = GetObject(directorio2)
Set objsheet1Plantilla = objexcelPlantilla.Sheets.Item(1)
Set objsheet2Plantilla = objexcelPlantilla.Sheets.Item(2)
Set objsheet3Plantilla = objexcelPlantilla.Sheets.Item(3)
```

Código 3.18 Definición de la plantilla Excel de referencia.

3. Variación en las entradas de ciertas funciones.
   - *ParametrosGenerales* y *ArbolModelado*: La ventana de CATIA, el documento, el *product* y el *part* de la referencia se dejan en blanco.
   - *AnalisisOperaciones*: a la hora de definir *mat1*, se deja en blanco la casilla correspondiente a la plantilla de la referencia. Por otro lado, al definir *mat2*, se deja en blanco el documento y el *part* de la referencia.

```
mat2 = analisisOper ("", "", objsheet2, objsheet2Plantilla,
Opción1, indicador, 1)
```

Código 3.19 Matriz de referencia que devuelve el módulo AnalisisOper.

4. Empleo de la subfunción *Nivel3Plantilla* para establecer el nivel 3 de análisis de la referencia.

Quitando estas variaciones, la metodología que emplea el módulo es la misma que la de *Principal*.

3.2.4 ParametrosGenerales

Es una función que extrae los parámetros generales de la pieza, que aparecen en la herramienta de CATIA *Measure Inertia*. Adicionalmente, calcula si existen discrepancias entre los valores de la pieza y de la referencia, devolviendo la nota asignada tras el nivel uno de análisis.

Los parámetros que se van a medir se recogen en la *Tabla 3.3*.

Tabla 3.3 Parámetros generales que CPIT evalúa.

Tipo	Parámetro	Metodología de corrección	Técnica de verificación
Global	Posicionamiento en sistema de referencia	Obtención de coordenadas del *Bounding Box* de la pieza y de la referencia	Sustracción entre coordenadas de modelos. Correcto:0; Incorrecto: $\neq 0$
Global	Volumen	Obtención del volumen computado en la aplicación	Sustracción entre cifras de modelos. Correcto:0; Incorrecto: $\neq 0$
Global	Centro de masas	Consultar coordenadas del centro de gravedad	Ídem anterior
Global	Momentos de inercia	Consultar valores numéricos	Ídem anterior

Por tanto, se obtienen diez parámetros que nos servirán para comparar la pieza y la referencia. El peso de cada uno de ellos en la nota final es:

- Cada coordenada del *Bounding Box* vale 0.6 puntos.
- El volumen vale un punto.
- Cada coordenada del centro de gravedad vale 0.2 puntos.
- Cada momento de inercia vale 0.2 puntos.

Antes de mostrar el código de la función, se van a introducir las entradas que recibe:

- Las ventanas de CATIA de la pieza y de la referencia.
- El documento, el *part* y el *product* de la pieza y de la referencia.
- La hoja 1 del archivo Excel de resultados.
- La hoja 1 de la plantilla de referencia (para el caso en el que no se disponga del archivo *.CATPart* de la referencia).
- La variable *indicador* para saber si el usuario dispone de pieza de referencia.

Al principio del módulo se definen dos vectores, *vectorDatos* y *ValoresRef*, donde se irán almacenando los parámetros de la pieza y de la referencia respectivamente. Posteriormente, se definen las variables *pesoBBL*, *pesoVolumen* y *pesoCOGyMI*, que almacenan la puntuación que recibe cada uno de los parámetros a analizar. A continuación, se lleva a cabo un bucle *For*, que hará dos

iteraciones. En la primera, se extraen los datos de la pieza, mientras que en la segunda se extraen los de la referencia.

Dentro del bucle, mediante el comando *If* se ve si el usuario dispone de pieza de referencia mediante la variable *indicador* previamente definida. En caso de que tenga, los parámetros de la referencia se extraen del mismo modo que los de la pieza. Sin embargo, si no dispone de ella, CPIT accede a la hoja 1 de la plantilla de referencia y lee las celdas donde se almacenan los parámetros que se buscan.

A continuación se describen los códigos empleados para la obtención de los parámetros. Como son los mismos para la pieza y para la referencia, solo cambia el vector donde se almacenan los resultados (*vectorDatos* o *ValoresRef*), se adjuntarán solo los de la pieza.

Posicionamiento en Sistema de Referencia

Se pretende conocer las coordenadas del *Bounding Box* del modelo, dando como resultado las dimensiones del paralelepípedo en el que queda confinada la pieza. Se usa el comando *GetItem* que permite acceder a los parámetros y obtener sus valores. Es necesario que se haya aplicado en la pieza la herramienta *Measure Inertia* para que el comando se efectúe correctamente.

Por ejemplo, para obtener la coordenada x del *Bounding Box* en milímetros, se ha empleado el siguiente código. [7].

```
vectorDatos(1) = productDocument.Parameters.GetItem("BBLx").Value
```

Código 3.20 Obtener la coordenada x del Bounding Box.

Volumen

Para obtener el volumen de la pieza se crea una referencia del *Body* del *Part*. Posteriormente se crea un *Workbench* del documento activo de CATIA, y se finaliza obteniendo la medida de la referencia en el *Workbench* mediante el comando *GetMeasurable*.

Este método será usado en distintos módulos de CPIT, ya que es la forma de obtener la referencia de un elemento de la pieza y medirla, extrayendo parámetros que serán determinantes para valorar el modelo generado por el usuario.

El código empleado es:

```
Dim objSPAWkb As Workbench
Dim objRef As Reference
Set objRef = partDocument.CreateReferenceFromObject(partDocument.
    MainBody)
Set objSPAWkb = documento.GetWorkbench("SPAWorkbench")
Set objMeasurable = objSPAWkb.GetMeasurable(objRef)
```

Código 3.21 Crear y medir una referencia de un objeto.

Coordenadas centro de gravedad

En primer lugar se define la variable *oInertia* como la inercia del *product* del documento activo en CATIA con el siguiente código, [7].

```
Set oInertia = productDocument.GetTechnologicalObject("Inertia")
```

Código 3.22 Obtener la inercia del modelo activo en CATIA.

Además, se establece un vector denominado *COG* de longitud tres donde se almacenan las coordenadas del centro de gravedad:

```
oInertia.GetCOGPosition COG
```

Código 3.23 Obtener el centro de gravedad de la pieza.

Por útlimo, se iguala *COG* a las posiciones cinco, seis y siete del vector *vectorDatos*.

Momentos Principales de Inercia

Para obtener los momentos principales de inercia se usa el objeto *oInertia* definido previamente. Se declara un vector de longitud tres denominado *Momentos*, y se almacenan en cada una de las posiciones los tres momentos con el siguiente código.

```
oInertia.GetPrincipalMoments Momentos
```

Código 3.24 Obtener los momentos principales de inercia de la pieza.

Por último, se iguala el vector Momentos a las posiciones ocho, nueve y diez del vector *vectorDatos*.

Si en lugar de disponer del archivo *.CATPart* de la referencia se dispone del archivo *.xlsx*, CPIT analiza la hoja 1 de la plantilla de Excel, almacenando los parámetros que va leyendo en el vector *ValoresRef*.

```
For k = 1 To 10
    ValoresRef(k) = objsheetPlantilla.Cells(k + 5, 4)
Next
```

Código 3.25 Definición del vector ValoresRef si la referencia viene dada por un archivo *.xlsx*.

Una vez entendido cómo obtener las medidas de la pieza y de la referencia, se pasa a ver las discrepancias entre las mismas.

Discrepancias

En primer lugar, se define un vector denominado *discrepancias* donde se almacenarán los resultados de las diferencias realizadas, y otro denominado *comentarios* donde se analizan las discrepancias obtenidas. Además, se define un error máximo de 1e-10, que es la tolerancia del fallo. Las diferencias llevadas a cabo serán iguales en todas las medidas previamente realizadas. La estructura es simple. Primero se establece en valor absoluto la diferencia entre el parámetro de la referencia y de la pieza. Posteriormente, mediante el comando *If*, se ve si esta diferencia es mayor o menor al error

máximo previamente definido. En caso de ser mayor, se almacena la palabra *Incorrecto* en el vector *discrepancias* y los comentarios pertinentes en el vector *comentarios*. En caso de ser menor, se almacena la palabra *Correcto* en el vector *discrepancias* así como los comentarios oportunos en el vector *comentarios*. Además, si se detecta que existen discrepancias entre la pieza y la referencia, se va actualizando el valor de la nota de la pieza.

Una vez finalizado el análisis, se procede a exportar los resultados obtenidos a la hoja 1 del archivo Excel. Se define el vector *Parámetros*, donde se guardan los nombres y las unidades de las mediciones realizadas. Esta será la primera columna de la tabla de resultados. Posteriormente, se van escribiendo los vectores *vectorDatos*, *ValoresRef*, *discrepancias* y *comentarios* en columnas adyacentes. Si se desea ver cómo queda la hoja de Excel, puede consultar el *Capítulo 4*.

Para finalizar la función, se devuelve el valor actualizado de la nota al módulo *Principal*.

3.2.5 ArbolModelado

Es una función encargada de extraer el árbol de modelado y evaluar el orden de las operaciones en el modelo, contándolas y comparándolas para comprobar si coinciden en la pieza y la referencia. Devuelve el valor de la puntuación asignada a este módulo (consultar *Figura 3.9*), que equivale a un 30 % de la nota final. Las entradas de esta función son:

- Las ventanas de CATIA de la pieza y de la referencia.
- El documento, el *part* y el *product* de la pieza y de la referencia.
- La hoja 2 del archivo Excel de resultados.
- La hoja 2 de la plantilla de referencia (para el caso en el que no se disponga del archivo *.CATPart* de la referencia).
- *ListBox1*, que es la ventana de resultados del menú principal.
- La variable *indicador* para saber si el usuario dispone de pieza de referencia.

En primer lugar, se declaran una serie de variables. En las matrices *matrizDatos* y *ValoresRef* se almacenan el nombre y el tipo de las operaciones empleadas para realizar la pieza y la referencia, respectivamente. Por otro lado, en el vector *numOperac* se guarda en la primera posición el número de operaciones de la pieza y en la segunda posición el de la referencia. Las matrices *mat* y *mat2* se usan para contar el número total de operaciones que hay de cada tipo en el árbol de modelado. Para ello se usan las variables *fil*, *fil2* y *cont*, que se explicarán más adelante. Por último, se declaran los valores *notaArbol* y *notaOrden*, que se igualan a dos y a uno respectivamente.

Una vez definidas las variables, se seleccionan y se cuentan las operaciones del árbol de modelado. El código empleado es el siguiente:

```
Dim seleccion As Selection
Set seleccion = documento.Selection
seleccion.Search "'Part Design'.'PartDesign Feature' + 'Part Design
    '.'Multi-sections Solid',all"
numOperac(0) = seleccion.Count
```

Código 3.26 Selección de las operaciones del árbol de modelado.

CPIT solo selecciona las operaciones del tipo *Mechanical Feature* y *Multi-Sections Solid*. Posteriormente se seleccionan los elementos de referencia empleados por el usuario para modelar la pieza, como puntos, líneas o planos. Sin embargo, estos no tendrán peso en la puntuación final.

Se va a explicar cómo se almacenan las operaciones de la pieza. Cuando se tienen el número de operaciones de la pieza, se pasa a definir las posiciones de la matriz *matrizDatos*. Lo primero que se hace es asignarle un número de filas igual al valor almacenado por *numOperac(0)*:

```
ReDim matrizDatos(numOperac(0), 1)
```

Código 3.27 Redefinir la matriz matrizDatos.

Posteriormente, mediante un bucle *For*, se almacena el nombre de las operaciones en la primera columna y el tipo de las mismas en la segunda columna. Por último, se define la matriz *mat* al igual que *matrizDatos* y se deselecciona el árbol de modelado de la pieza:

```
seleccion.Clear
```

Código 3.28 Borrar la selección previamente realizada.

Una vez analizada la pieza, se pasa a la referencia. Como se sabe, pueden darse dos escenarios: que exista pieza de referencia o que no exista. En caso afirmativo, la obtención de las operaciones sería igual a la de la pieza, cambiando la matriz *matrizDatos* por *ValoresRef*. En caso contrario, dado que las operaciones estarían almacenadas en la hoja 2 de la plantilla de Excel, se realizarían una serie de iteraciones para leer las celdas con información. El funcionamiento del bucle es sencillo. Primero se iguala el número de operaciones a cero, ya que se irá incrementando a medida que se detecte que las celdas no están vacías. Cuando se llegue a la primera celda carente de información, se sale del bucle *For*.

```
If objsheetPlantilla.Cells (k + 5, 2) <> "" Then
    numOperac (1) = numOperac (1) + 1
ElseIf objsheetPlantilla.Cells (k + 5, 2) = "" Then
    Exit For
End If
```

Código 3.29 Definición del número de operaciones de la referencia cuando no se dispone de su archivo .CATPart.

Por último, se definen las dimensiones de *ValoresRef* y de *mat2*, y se almacena en la primera el nombre y el tipo de las operaciones que conforman la referencia.

Una vez se han completado las matrices de operaciones, se pasan a analizar los elementos que CPIT no tiene en cuenta a la hora de evaluar el programa. Estos son:

- Puntos, líneas y planos que el usuario use de referencia.
- Traslaciones, rotaciones y simetrías empleadas.
- Sketches que estén en el árbol de modelado.

El módulo define dos matrices llamadas *operacionesApoyo* y *operacionesApoyoRef*, asociadas a la pieza y a la referencia respectivamente. Ambas tienen siete filas y dos columnas, guardándose en la

primera el nombre de la operación y en la segunda el número que hay de la misma. Por ejemplo, para buscar las traslaciones de la pieza en el árbol de modelado se emplea el código:

```
seleccion.Search "'Part Design'.'Translate',all"
```

Código 3.30 Selección de las traslaciones en el árbol de modelado.

Solo se tendría que cambiar *Translate* por la operación que se desee encontrar. En el caso de no disponer de pieza de referencia, se realiza un bucle *For* para leer y extraer los parámetros almacenados en la plantilla de resultados. [7].

A continuación, la función pasa a exportar los resultados empleando un bucle *For* con tantas iteraciones como operaciones tenga la pieza. CPIT escribe en la hoja 2 de Excel tanto el nombre como el tipo de la pieza y de la referencia. Consultar el *Capítulo 4* si se desea ver cómo queda.

CPIT indica si las operaciones de la pieza y de la referencia coinciden con el mensaje:

CORRECTO: El número de operaciones de la pieza y de la referencia coinciden

En caso contrario, el sombreado del texto sería rojo y el mensaje diría:

INCORRECTO: El número de operaciones de la pieza y de la referencia no coinciden

Para llegar a esta conclusión, se han comparado el tipo de operaciones y el número de cada una de ellas. Para ello, se definen las matrices *mat* y *mat2*, asociadas a la pieza y a la referencia respectivamente, que almacenarán en la primera columna el tipo y en la segunda el número de cada tipo.

Se realiza un análisis iterativo de las matrices *matrizDatos* y *ValoresRef*, más concretamente de la segunda columna de las mismas, donde se almacenaba el tipo de la operación. El método empleado se resume en el diagrama de la *Figura 3.10*.

La variable *fil* se emplea para saber cúantas filas no están vacías. Si se dispone de pieza de referencia, la matriz *mat2* se completa de la misma forma, pero empleando las variables *ValoresRef* y *fil2*. En caso contrario, se realiza un bucle *For* para leer las celdas donde está almacenada la información. Si se detecta una celda vacía, se sale del bucle. Una vez completadas las matrices, se escriben los resultados en la hoja 2 del archivo Excel. Adicionamente, se escriben los elementos de referencia que previamente se han analizado.

A continuación, se comprueba que la pieza y la referencia tengan el mismo número de operaciones. Se pueden dar diferentes casos:

- **fil** $\neq$ **fil2**: las operaciones de la pieza y de la referencia no coinciden. Se almacena en la variable *text* el mensaje: *"INCORRECTO: El número de operaciones de la pieza y de la referencia no coinciden"*, y se muestra sombreado en rojo en la hoja 2 del archivo Excel. Por último, se muestra por la ventana de resultados el mensaje:

  -INCORRECTO: Las operaciones no coinciden con las de la referencia.

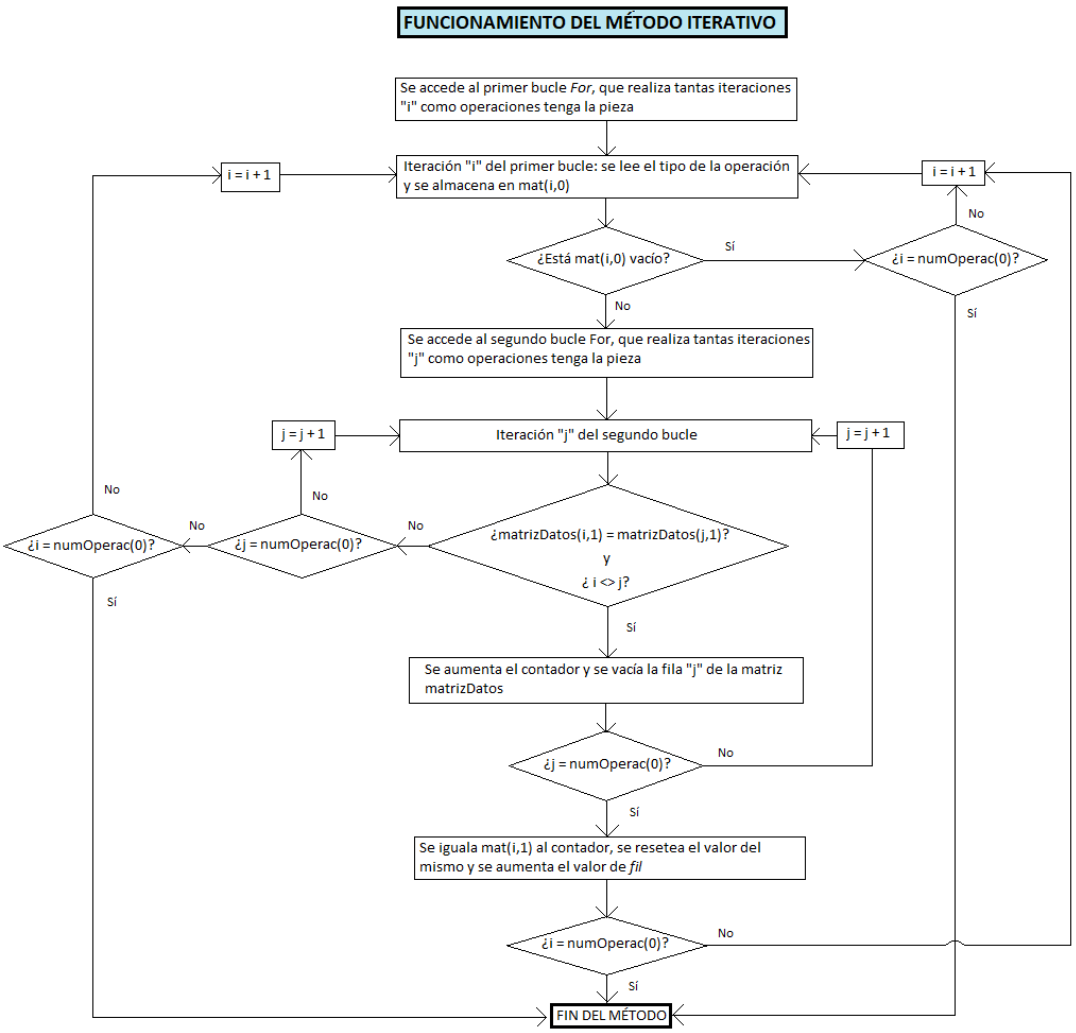


Figura 3.10 Diagrama de flujo sobre cómo obtener *mat*.

- **fil = fil2, pero no coincide el contenido de las matrices**: las operaciones de la pieza y de la referencia no coinciden. Se realiza un barrido de las filas de *mat* y *mat2* para ver si hay discrepancias en el número de operaciones de cada tipo. Al detectarse un error, se guarda en la variable *text* el mismo mensaje que en el caso anterior, además de mostrarse la misma información por la ventana de resultados.
- **fil = fil2, coincidiendo el contenido de las matrices**: se analiza la variable *text*. Si está vacía, significa que las matrices son idénticas, por lo que se almacena en la variable *text* el mensaje: *"CORRECTO: El número de operaciones de la pieza y de la referencia coinciden"*. Además, se muestra por pantalla el siguiente mensaje:

-CORRECTO: El número de operaciones coincide con el de la referencia.

Analizadas las discrepancias existentes entre ambas piezas, el módulo pasa a actualizar los valores de *notaArbol* y *notaOrden*. En primer lugar, se actualiza el parámetro *notaArbol*. Para ello, se ha empleado el siguiente método de corrección:

1. Se accede a dos bucles *For* que analizan las celdas de la hoja 2 de Excel en las que se almacenan el tipo y la cantidad de cada operación. El objetivo es comparar cada tipo de la pieza del usuario con todas las de la referencia, para comprobar si dicha operación se encuentra en ambas piezas.

2. Dentro del bucle *For* de la referencia pueden darse dos casos:

- La celda que se analiza está vacía, lo que provoca que el programa se salga del bucle.
- El tipo de la operación de la pieza y de la referencia coincida. En este caso, se calcula la diferencia entre la cantidad de cada una, pudiéndose dar tres casos:
  - diferencia < 0: la referencia tiene un mayor número de esa operación que la pieza. En este caso, se actualiza la nota como:

    notaArbol = notaArbol - Abs(dif) * 2 / numOperac (1)
  - diferencia > 0: la referencia tiene un menor número de esa operación que la pieza. En este caso, se actualiza la nota como:

    notaArbol = notaArbol - Abs(dif) * 0.1
  - diferencia $= 0$: tanto la pieza como la referencia tienen el mismo número de esa operación.

3. Pasado el bucle de la referencia, se comprueba si tanto la celda de la pieza como la de la referencia se encuentran vacías. En caso afirmativo, el programa se sale del bucle de la pieza, dándose por concluida la actualización del parámetro *notaArbol*.

4. En caso de no haberse encontrado la operación de la pieza en la referencia, se analiza si hay alguna similar o si es totalmente distinta a las existentes.

- Si existe alguna operación similar, la nota se actualiza como:

  notArbol = notaArbol - 0.15
- Por contra, si es totalmente distinta al resto se acutaliza como:

  notArbol = notaArbol - 0.3

5. En caso de que la celda seleccionada esté vacía, pero la de la referencia no, la nota se actualiza como:

notaArbol = notaArbol - (fil2 - contIg) * 0.3

Donde *contIg* es el número de operaciones de la pieza que coinciden con las de la referencia. De esta forma se le resta 0.3 por cada operación que le falte al modelo del usuario.

Actualizado el parámetro *notaArbol*, se pasa a hacer lo mismo con *notaOrden*. En este caso, se determina que cada fallo resta *1/numOperac(0)*. Se considera fallo a que la operación de la pieza no se encuentre en la misma posición del árbol de modelado que la de la referencia. Tanto *notaArbol* como *notaOrden* tienen que ser siempre mayores o iguales a cero, por lo que si son negativas se fuerza a que cumplan esta condición.

Para finalizar el módulo, se devuelve a *Principal* el valor:

arbolMod = notaArbol + notaOrden

3.2.6 ContadoresOper

Se trata de una función que cuenta el número de operaciones que hay de cada tipo en la pieza, necesitando solo de entrada la hoja 2 del archivo Excel de resultados. La metodología que emplea se basa en leer las celdas de las tablas en las que se guardan el tipo y la cantidad de las operaciones que conforman la pieza y la referencia. El funcionamiento del módulo se presenta en el diagrama de la *Figura 3.11*.

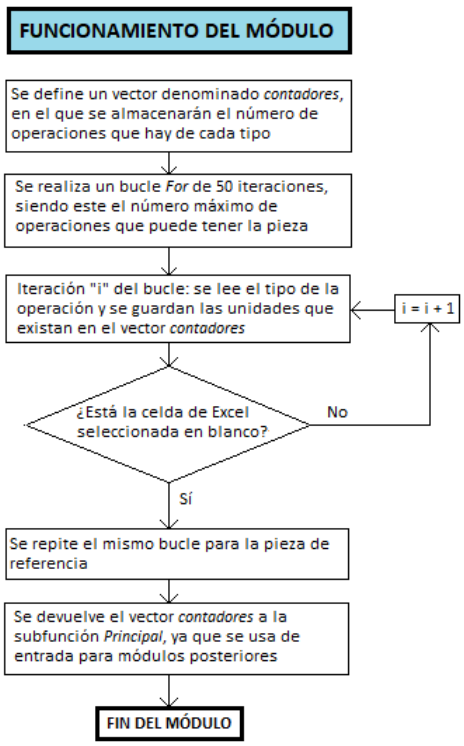


Figura 3.11 Diagrama de flujo sobre cómo contar las operaciones que hay de cada tipo.

Como se puede observar en el diagrama adjunto, se define un vector denominado *contadores*, en el que se guardan el número de operaciones que hay de cada tipo tanto en la pieza del usuario como en la de la referencia. Tiene 32 posiciones definidas por:

- Primera posición: número de *pad* en la pieza del usuario.
- Segunda posición: número de *hole* en la pieza del usuario.
- Tercera posición: número de *pocket* en la pieza del usuario.
- Cuarta posición: número de *rib* en la pieza del usuario.
- Quinta posición: número de *slot* en la pieza del usuario.
- Sexta posición: número de *rectangular pattern* en la pieza del usuario.
- Séptima posición: número de *multi-sections solid* en la pieza del usuario.
- Octava posición: número de *circular pattern* en la pieza del usuario.
- Novena posición: número de *edge fillet* en la pieza del usuario.
- Décima posición: número de *shell* en la pieza del usuario.
- Undécima posición: número de *draft* en la pieza del usuario.
- Duodécima posición: número de *chamfer* en la pieza del usuario.
- Decimotercera posición: número de *groove* en la pieza del usuario.
- Decimocuarta posición: número de *stiffener* en la pieza del usuario.
- Decimoquinta posición: número de *shaft* en la pieza del usuario.
- Decimosexta posición: número de *mirror* en la pieza del usuario.

El resto de posiciones quedan definidas de la misma forma, pero en lugar de la pieza del usuario se analiza la pieza de referencia.

3.2.7 OperacionesIguales

Se trata de una función que extrae en una matriz llamada *matOperIguales* las operaciones que pertenecen al árbol de modelado tanto de la pieza como de la referencia. Este módulo determina cuáles son las operaciones que CPIT va a analizar.

Las entradas de esta subfunción son:

- Las hoja 2 del archivo Excel de resultados.
- La variable *PiezaOReferencia*, que será cero si se quiere analizar la pieza del usuario y uno si se quiere analizar la referencia.

El funcionamiento del módulo se presenta en el diagrama de la *Figura 3.12*.

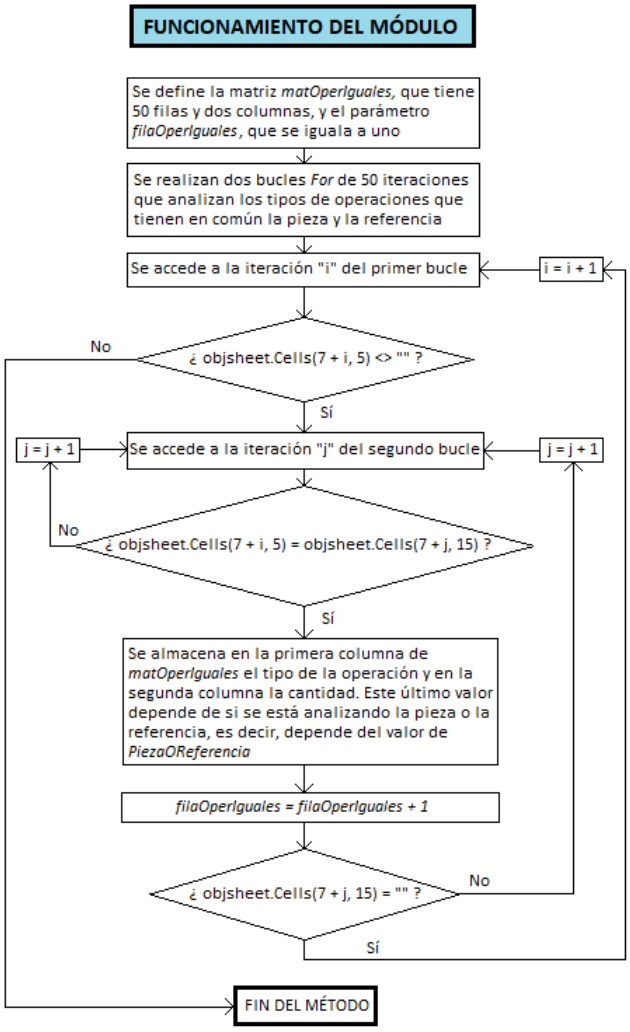


Figura 3.12 Diagrama de flujo sobre cómo obtener la matriz *matOperIguales*.

Una vez definida la matriz *matOperIguales* se analiza cuántas de sus filas contienen información, guardándose el resultado en una variable llamada *numOperIguales*. Tanto este valor como la matriz se devuelven al módulo *Principal* o *Principal2*.

3.2.8 AnalisisOperaciones

El objetivo de esta función es obtener una matriz con los parámetros tridimensionales de las operaciones. Se le llamará desde el módulo *Principal* o *Principal2* dos veces, una para la pieza y otra para la referencia, devolviendo dos matrices que se usarán como entrada en los módulos *Comparar* y *PintarResultadosAnalisisOper*. Las entradas de esta función son:

- El documento y el *part* de la pieza que se analice.
- La hoja 2 del archivo Excel de resultados.
- La hoja 2 de la plantilla de referencia (para el caso en el que no se disponga del archivo *.CATPart* de la referencia).
- La variable *indicador* para saber si el usuario dispone de pieza de referencia.
- La variable *PiezaOReferencia*, que será cero si se quiere analizar la pieza del usuario y uno si se quiere analizar la referencia.
- La matriz *matOperIguales* y el vector *contadores* previamente hallados.

En primer lugar, se define la variable *matr*, que es la matriz donde se almacenan los resultados que se van obteniendo. Posteriormente, se define el parámetro *colum*, que se usa para seleccionar la columna del archivo Excel de resultados en la que exportar los datos. En el caso de la pieza toma el valor dos, mientras que en el caso de la referencia toma el valor doce. Antes de obtener las dimensiones tridimensionales de las operaciones que conforman la pieza, se realizan una serie de pasos previos.

En primer lugar, se cuentan las filas que ocupa la tabla asociada a los elementos de referencia, que se definieron previamente. El objetivo es hacer una hoja de resultados dinámica, es decir, que las tablas siempre mantengan las mismas celdas de separación independientemente de la información que almacenen. Con ello se mejora la estética del archivo de resultados y se evita que las tablas se solapen entre sí. Para realizar este conteo, se accede a la hoja 2 de resultados y se busca la palabra *Elemento* en la columna dos.

```
    If objsheet.Cells(n,2) = "Elemento" Then
```

Código 3.31 Búsqueda de la palabra "Elemento" en la segunda columna de la hoja 2 de Excel.

Una vez se encuentre, se almacena la fila de dicha celda en la variable *fila*. Posteriormente, se recorre descendentemente la columna hasta que se encuentre una celda vacía. El bucle realiza siete iteraciones, ya que es el máximo número de elementos de referencia que puede tener la pieza. Además, se busca que estén vacías tanto la celda de la columna dos como la de la columna diez, asociadas a la pieza y a la referencia respectivamente. De esta forma evitamos problemas en el caso de que los elementos de referencia de ambas operaciones sean distintos. Detectadas ambas celdas, se almacena la iteración correspondiente en la variable *filasTabla2* y el programa se sale del bucle.

```
    If objsheet.Cells(fila+m,2) = "" And objsheet.Cells(fila+m,12) = ""
       Then
       filasTabla2 = m
       Exit For
    End If
```

Código 3.32 Definición de la variable filasTablas2.

El último paso previo al análisis tridimensional es definir el parámetro *numOper*, al que se le asigna el número de operaciones de la pieza que tenga más. Este valor se emplea para el ajuste dinámico de las tablas en la hoja de resultados.

En lo que sigue, el módulo va a extraer las dimensiones 3D de las operaciones almacenadas en *matOperIguales*. Para ello, se van a emplear dos métodos distintos dependiendo de lo que se esté analizando:

- **Método 1**: es empleado tanto en la pieza y en la referencia cuando se dispone del archivo *.CATPart* de esta última, así como en la pieza cuando no se dispone del mismo.
- **Método 2**: analiza las operaciones de la referencia cuando no se dispone del *.CATPart*.

A continuación se exponen una serie de procesos comunes para ambos métodos. Primero, el módulo declara una serie de variables, que están asociadas a las operaciones que posteriormente se analizarán. Entre las variables, cabe destacar la denominada *fil*. Esta será de ayuda para saber en qué fila de la hoja de resultados se escribe el título de la función: *ANÁLISIS 3D DE LAS OPERACIONES*. Además, me ayuda a definir la variable *separacionTablas*:

```
separacionTablas = fil + 4
```

Código 3.33 Definición de la variable separacionTablas.

Esta variable me indica dónde empieza la tabla asociada a la primera operación, con valores tridimensionales, de la pieza y de la referencia. Cuando se rellene la tabla, se actualiza dicha variable para que almacene la fila en la que deberá comenzar la siguiente tabla, y así sucesivamente.

Antes de empezar el análisis tridimensional, se definen una serie de contadores e iteradores para cada una de las operaciones. Los contadores almacenarán el número que hay de cada tipo y los iteradores irán aumentando al detectar una operación analizada previamente. La función de estos últimos es saber la fila en la que escribir la información asociada a cada operación del mismo tipo. Todos los iteradores empiezan valiendo uno, mientras que los contadores se obtienen como entrada de la función.

Por último, se definen las variables *indOperIguales* y *filaMAT*. La primera se emplea para indicar si la operación seleccionada en el árbol de modelado se encuentra en *matOperIguales*, mientras que la segunda se emplea para indicar la fila de la matriz *matr* donde almacenar la información. La matriz *matr* tiene un número de filas igual a *numOperIguales* y diez columnas.

Se va a comenzar explicando el Método 1, al que se accede mediante el código:

```
If indicador <> 1 Or PiezaOReferencia <> 1 Then
```

Código 3.34 Condiciones comando If módulo AnalisisOperaciones.

A continuación, se seleccionan las operaciones del árbol de modelado y se cuentan, almacenando el resultado en la variable *numOperDeLaPieza*. El módulo comienza el análisis de las operaciones empleando un bucle *For* que va desde uno hasta este valor. El método es el siguiente:

1. Se selecciona la operación asociada a la iteración en la que se está, y se guarda su nombre en la variable *nombreOper* y su tipo en *tipoOper*.

```
nombreOper = seleccion.Item(i).Value.name
tipoOper = seleccion.Item(i).Type
```

Código 3.35 Definición de las variables nombreOper y tipoOper.

2. Se comprueba que la operación seleccionada se encuentra en *matOperIguales*. En caso negativo, no se analiza esta operación y se selecciona la siguiente del árbol de modelado.
3. En caso afirmativo, se pasa a completar las distintas filas de *matr*. Mediante el comando *If* se analiza el contenido de *tipoOper*. Una vez se sabe el tipo de la operación, almacenamos su valor. Por ejemplo, en el caso del *pad*:

```
If tipoOper = "Pad" Then
    pad1 = seleccion.Item(i).Value
```

Código 3.36 Almacenar el valor de una operación.

4. Se extraen todos los valores tridimensionales asociados a esa operación y se almacenan en la fila correspondiente de *matr*.
5. Se vuelve al primer paso hasta terminar de analizar todo el árbol de modelado de la pieza.

Es interesante saber los valores que CPIT es capaz de extraer:

- ***Pad***: altura (mm).
- ***Hole***: tipo, diámetro (mm), profundidad (mm), extensión (*blind*, *up to next*, etc.), diámetro superior (mm), profundidad superior (mm), ángulo (º).
- ***Pocket***: profundidad (mm).
- ***Rectangular Pattern***: nº de repeticiones, objeto copiado.
- ***Circular Pattern***: nº de repeticiones, objeto copiado.
- ***Mirror***: objeto copiado, plano de referencia, coordenadas (mm) de los vectores directores que forman el plano de referencia.
- ***Edge Fillet***: radio (mm), nº de objetos redondeados, longitudes (mm) de los *edges* que han sido redondeados.
- ***Stiffener***: espesor (mm).
- ***Chamfer***: ángulo (º), longitud (mm), número de elementos seleccionados.
- ***Draft***: ángulo (º), nº de caras seleccionadas, coordenadas (mm) de los vectores directores que forman el *Neutral Element*.
- ***Shaft***: primer ángulo (º), segundo ángulo (º), eje de revolución.
- ***Groove***: primer ángulo (º), segundo ángulo (º), eje de revolución.
- ***Shell***: espesor externo (mm), espesor interno (mm), nº de caras seleccionadas.
- ***Multi-Sections Solid***: nº de guías empleadas, nº de sketches seleccionados, nº de contornos.
- ***Rib***: no tiene.
- ***Slot***: no tiene.

En el caso de que el usuario no disponga de la pieza de referencia, sino de una plantilla de Excel con los datos de la misma, el método de obtención de *matr* es distinto (Método 2). Como no se dispone del archivo *.CATPart*, en lugar de seleccionar las operaciones del árbol de modelado, se accede a la hoja 2 de la plantilla de resultados. A continuación, se leen las operaciones que conforman la referencia y se almacenan en *matr* solo aquellas que coincidan tanto en la pieza como en la referencia.

Para finalizar, el módulo devuelve la matriz de resultados a *Principal* o *Principal2*.

3.2.9 PintarResultadosAnalisisOper

Este módulo tiene como finalidad exportar los datos almacenados por las matrices *mat1* y *mat2* obtenidas mediante la función *AnalisisOperaciones* a la hoja 2 del archivo de resultados.

Las entradas de esta subfunción son:

- Las matrices *mat1* y *mat2*.
- La hoja 2 del archivo Excel de resultados.
- El vector *contadores* previamente hallado.

En primer lugar se vuelven a definir algunos parámetros que fueron de utilidad en el módulo anterior. Estos son:

- *numOper*, *filasTabla2*, *fil* y *separacionTablas*.
- Los contadores e iteradores de las operaciones que conforman tanto la pieza como la referencia.

A continuación se realiza un bucle de dos iteraciones, usando la primera para exportar los valores de *mat1* y la segunda para exportar los valores de *mat2*. Antes de comenzar a escribir los resultados en la hoja de Excel, se definen una serie de variables cuyos valores dependen de la iteración en la que se encuentre el programa:

- *colum*: se iguala a dos para la pieza y a doce para la referencia. Indica la columna que se toma de referencia para escribir los resultados.
- *iter*: establece el número de filas de la matriz que se analiza (*mat1* o *mat2*). Se define mediante el comando:

```
UBound (matriz, 1) - LBound (matriz, 1)
```

Código 3.37 Contar las filas de una matriz.

- *matr*: se iguala a la matriz que se analiza (*mat1* o *mat2*)).
- Por último, se escogen los contadores de las operaciones que conforman la pieza correspondientes a la iteración en la que se encuentre el programa.

Definidas todas las variables necesarias para el correcto funcionamiento del módulo, se realiza un bucle *For* de un número de iteraciones igual al valor de *iter*. Dentro del bucle, se identifica el tipo de cada una de las operaciones que conforman la matriz *matr*, y se escriben todos sus parámetros característicos en la hoja dos del archivo de resultados. En cada iteración, se actualiza el valor de *separacionTablas* y de los iterantes de cada operación, evitando posibles solapamientos en la hoja de Excel.

3.2.10 Nivel3

Este módulo se encarga de extraer los parámetros bidimensionales de los *sketches* de las operaciones. Es una subfunción, por lo que no va a devolver ningún resultado. Su objetivo es obtener las

dimensiones de los elementos geométricos que conforman las operaciones y exportar los resultados a la hoja 3 de Excel.

Esta subfunción se llama dos veces, una para la pieza y otra para la referencia, en el caso de tener el archivo *.CATPart* de esta última. En el caso contrario, solo es llamada para la pieza, usando el módulo *Nivel3Plantilla* para el análisis bidimensional de la referencia, que se explicará posteriormente.

Las entradas de esta subfunción son:

- El documento de la pieza que se analice.
- Las hojas 2 y 3 del archivo Excel de resultados.
- La variable *indicador* para saber si el usuario dispone de pieza de referencia.
- La variable *PiezaOReferencia*, que será cero si se quiere analizar la pieza del usuario y uno si se quiere analizar la referencia.
- La matriz *matOperIguales* previamente hallada.

El comienzo del módulo es parecido al de la función *AnalisisOperaciones*. Se define la variable *colum*, que será de ayuda para exportar las medidas a Excel. A continuación, se seleccionan las operaciones del árbol de modelado de la pieza y se cuentan, almacenando el resultado en la variable *num1*. Seguidamente, se definen una serie de variables empleadas para el correcto funcionamiento de la subfunción. Entre ellas, cabe destacar los contadores *contPuntos*, *contLineas* y *contCirc*, que cuentan el número de puntos, líneas y circunferencias que hay en el sketch, respectivamente. Además, el módulo declara una matriz llamada *mat*, que tendrá treinta y una columnas y un número de filas igual al parámetro *numOperIguales*. En la primera columna se almacena el nombre de la operación, en las tres siguientes las variables *contPuntos*, *contLineas* y *contCirc* asociadas al primer sketch de la misma, y en las restantes se guardan las mismas variables pero asociadas al resto de *sketches* de la operación. En el caso de que solo exista un *sketch*, estas columnas quedan vacías.

Adicionalmente, se definen tres matrices denominadas *matPuntos*, *matLineas* y *matCirc*, que se emplean para almacenar las dimensiones de cada punto, línea y circunferencia, respectivamente.

Las últimas variables declaradas son las llamadas *separacionTablas* y *separacionTablas2*, que serán de ayuda a la hora de exportar las mediciones a la hoja Excel de resultados. La primera establece las celdas de separación entre la información de cada sketch de cada operación, mientras que la segunda establece las celdas de separación entre la información de los elementos geométricos de un mismo sketch.

A continuación, el módulo pasa a realizar un bucle *For* de un número de iteraciones igual a la variable *num1*. En cada iteración, se almacena el nombre y el tipo de la operación en las variables *nombreOper* y *tipoOper*, respectivamente. Posteriormente, se determina si dicho tipo coincide con alguno de los almacenados en la matriz *matOperIguales*. En caso afirmativo, se selecciona la operación y se cuenta el número de *sketches* que tiene mediante el código:

```
seleccion2.Search "Name=" & nombreOper
seleccion3.Search "'Part Design'.'Sketch',sel"
contSketches = seleccion3.Count
```

Código 3.38 Contar el número de sketches de una operación.

Una vez identificados los *sketches* de la operación, se pasa a analizar cada uno de ellos. Lo primero que se hace es contar el número de elementos geométricos que hay en cada sketch. [7].

```
Set sketch1 = seleccion3.Item(j).Value
Set elementosGeo = sketch1.GeometricElements
numGeoElem = elementosGeo.Count
```

Código 3.39 Contar el número de elementos geométricos de un sketch.

El parámetro *elementosGeo* almacena una lista con todos los elementos geométricos del sketch seleccionado, por lo que para establecer el valor de *contPuntos*, *contLineas* y *contCirc* solo hay que inspeccionar la lista elemento por elemento. Por ejemplo, el código empleado para aumentar el contador de las líneas es:

```
Set elemGeo = elementosGeo.Item(k)
If elemGeo.GeometricType = catGeoTypeLine2D Then
    contLineas = contLineas + 1
```

Código 3.40 Contar el número de líneas de un sketch.

siendo *elemGeo* el elemento geométrico que se está analizando.

Hay que tener cuidado con las proyecciones que se realizan en los *sketches* con la herramienta *Project 3D Elements*, ya que CPIT las identifica como elementos geométricos desconocidos. Para solucionar este problema, se crea una referencia de la proyección, se mide y se emplea el comando *Measure.GeometryName*, donde se pueden dar tres casos:

- *Mea.GeometryName* = *CatMeasurableCircle*: la proyección es una circunferencia.
- *Mea.GeometryName* = *CatMeasurableLine*: la proyección es una línea.
- *Mea.GeometryName* = *CatMeasurablePoint*: la proyección es un punto.

Identificado el tipo del elemento, se procede a aumentar el contador correspondiente. Una vez definidos todos los contadores, se almacenan en la matriz *mat*. Este proceso se ha de repetir para cada uno de los *sketches* de la operación seleccionada.

A continuación, se evalúan los elementos uno a uno. Lo primero que se hace es establecer el tamaño de las matrices *matPuntos*, *matLineas* y *matCirc*.

```
ReDim matLineas (mat (filaMAT, 1 + contador), 1)
ReDim matPuntos (mat (filaMAT, 2 + contador), 3)
ReDim matCirc (mat (filaMAT, 3 + contador), 1)
```

Código 3.41 Redefinir las matrices matLineas, matPuntos y matCirc.

En la primera columna de cada una de ellas se almacena el nombre del elemento geométrico. En la segunda columna de *matLineas* se almacena la longitud de la línea en milímetros, en la segunda columna de *matCirc* se almacena el radio de la circunferencia en milímetros, y en las columnas restantes de *matPuntos* se almacenan las coordenadas de los puntos también en milímetros. La variable *filaMAT* me indica la fila en la que se encuentra la información de la operación seleccionada, mientras que la variable *contador* se emplea en el caso de que haya más de un *sketch*, aumentando de tres en tres cada vez que analiza uno.

Una vez definidas, CPIT identifica el tipo del elemento que se ha seleccionado, para saber los parámetros que debe evaluar.

- ***elemGeo.GeometricType = catGeoTypeLine2D***: si el elemento es una línea, el módulo obtiene el punto inical y el punto final de la misma con el código:

```
Set origen = elemGeo.StartPoint
Set Destino = elemGeo.EndPoint
```

Código 3.42 Obtención del punto inicial y final de una recta.

Para obtener las coordenadas de las variables *origen* y *Destino*, se definen los vectores *coordSP* y *coordFP* con el código:

```
origen.GetCoordinates (coordSP)
Destino.GetCoordinates (coordFP)
```

Código 3.43 Definición de los vectores coordSP y coordFP.

Para establecer la longitud de la línea, se calcula la distancia que hay entre ambos puntos.

```
Lenght = Sqr((coordFP(0) - coordSP(0))^2 + (coordFP(1) -
    coordSP(1))^2)
```

Código 3.44 Obtención de la longitud de una recta.

Se almacena el valor obtenido en *matLineas*.

- ***elemGeo.GeometricType = catGeoTypePoint2D***: si el elemento es un punto, se distinguen dos casos:

  – Punto asociado al centro de un *hole*.
  – Cualquier otro punto.

En el caso de que no sea el centro de un *hole*, se obtienen las coordenadas del punto con el código:

```
elemGeo.GetCoordinates coordenadas
```

Código 3.45 Obtención de las coordenadas de un punto.

El hecho de emplear un método distinto para el *hole* se debe a que el comando *GetCoordinates* obtiene las coordenadas del punto respecto al sistema de referencia del sketch, que en el caso del *hole* no suele coincidir con el de la pieza. Cuando en CATIA se realiza un *hole*, el sistema de referencia se establece en el punto de la pieza donde el usuario clica para realizar esta operación. Por tanto, para referenciarlo al sistema de la pieza, se crea una referencia del punto y se mide con el comando *GetMeasurable*.

```
Set referencia = oPart.CreateReferenceFromObject(punto)
Set TheSPAWorkbench = documento.GetWorkbench("SPAWorkbench")
Set Mea = TheSPAWorkbench.GetMeasurable(referencia)
Set pt = Mea
pt.GetPoint coords
```

Código 3.46 Obtención de las coordenadas del punto de un hole.

Por último, se almacenan las coordenadas en la matriz *matPuntos*.

- ***elemGeo.GeometricType = catGeoTypeCircle2D***: si el elemento es una circunferencia, se extrae su radio:

```
radio = elemGeo.Radius
```

Código 3.47 Obtención del radio de una circunferencia.

Se almacena el valor del mismo en *matCirc*.

- ***elemGeo.GeometricType = catGeoTypeUnknown***: hace referencia a las proyecciones. Una vez identificado el elemento geométrico de la proyección, se extraen sus parámetros tal y como se ha explicado anteriormente.

CPIT exporta las mediciones obtenidas a la hoja 3 del archivo Excel de resultados. Puede consultar el *Capítulo 4* para ver cómo queda esta hoja para distintas piezas.

Aunque parezca que el análisis de los *sketches* es muy mecánico y similar para todas las operaciones, hay que tener precaución con varias de ellas. Un caso es el del *rib* y del *slot*, donde se pueden dar tres casos distintos en la definición de sus *sketches*:

- **Caso 1**: los dos definidos dentro de la operación.
- **Caso 2**: uno definido dentro de la operación y el otro situado en el árbol de modelado.
- **Caso 3**: ambos en el árbol de modelado, fuera de la operación.

Por tanto, si se da el segundo o tercer caso, el método expuesto anteriormente fallaría, ya que CPIT identificaría que la operación tiene uno y cero *sketch* respectivamente. Consecuentemente, se realiza un proceso alternativo que se basa en:

- **Caso 2**: se identifica cuál es el *sketch* que no está definido dentro de la operación para posteriormente buscarlo en el árbol de modelado. El *rib* y el *slot* están definidos por dos elementos:
  - *Profile*. Para saber cuál es el *sketch* asociado a este elemento se usa el comando:

```
Set profileRib = rib1.Sketch
```

Código 3.48 Obtención del sketch del profile de un rib.

  - *Center Curve*. Para saber cuál es el *sketch* asociado a este elemento se usa el comando:

```
Set centerCurveRib = rib1.CenterCurve
```

Código 3.49 Obtención del sketch del center curve de un rib.

Con estos comandos se conocen cuáles son los *sketches* de la operación y por ende cuál se debe buscar en el árbol.

- **Caso 3**: se buscan directamente en el árbol de modelado los *sketches* que definen el *profile* y *center curve* con los comandos que se nombraron previamente.

Otra operación que altera la metodología empleada es *Multi-sections Solid*, ya que no siempre quedan todos los *sketches* definidos dentro de la operación. Además, puede haber dos formas de realizarla:

1. Unión de un cierto número de *sketches*.
2. Unión de contornos (*boundary*).

La primera forma sigue la metodología empleada hasta el momento, aunque hay que remarcar que CPIT solo puede analizar *Multi-sections Solid* de diez *sketches* como máximo. En cambio, con la segunda forma, hay que usar un método distinto. En primer lugar se buscan cuántos contornos hay en la operación con el comando:

```
seleccion3.Search "((FreeStyle.Extract + 'Generative Shape Design'.
   Extract) + 'Functional Molded Part'.Extract),sel"
```

Código 3.50 Selección de los contornos de un Multi-sections Solid.

Una vez seleccionados, se cuentan con el comando *.Count*. A continuación se realiza un bucle *For* de tantas iteraciones como contornos haya, y en cada iteración se crea una referencia del contorno correspondiente para extraer su área y su perímetro. Por último, se exportan los resultados a la hoja 3 de Excel.

3.2.11 Nivel3Plantilla

Este módulo solo se emplea en el caso de tener una plantilla de referencia en lugar de un archivo *.CATPart*. Su objetivo es exportar las dimensiones de los elementos geométricos de la referencia, almacenados en la hoja 3 de la plantilla, a la hoja 3 del archivo Excel de resultados. Las entradas de esta subfunción son:

- La matriz *mat2*, resultado de la función *AnalisisOperaciones* para la referencia.
- La hoja 3 del archivo Excel de resultados.
- La hoja 3 de la plantilla Excel de la referencia.

En primer lugar, se definen la variables *cont* y *filaHoja*. La primera almacena el número de filas de la matriz *mat2*, mientras que la segunda se iguala a cuatro e indica la fila de la hoja 3 de la plantilla de la que se extrae información. Su valor se actualiza durante la ejecución del módulo.

Dado que en la plantilla aparecen los *sketches* de todas las operaciones de la referencia, el objetivo de este módulo es exportar a la hoja 3 de resultados solo aquellas operaciones que pertenecen tanto

a la pieza del usuario como a la referencia. La metodología empleada se recoge en el siguiente diagrama de flujo.

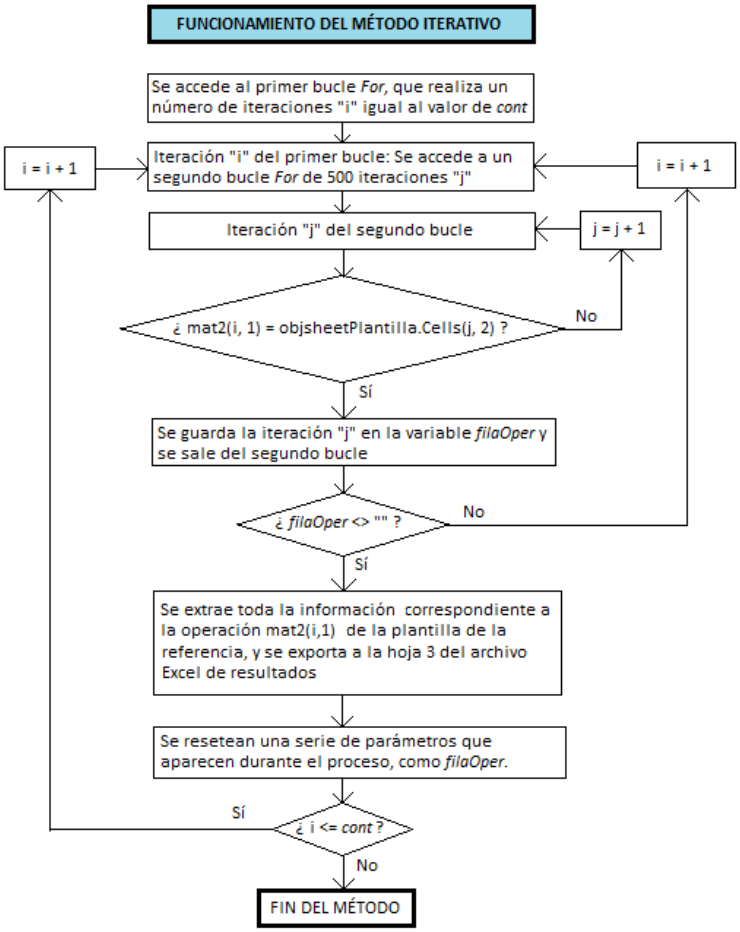


Figura 3.13 Diagrama de flujo sobre la metodología de *Nivel3Plantilla*.

3.2.12 Comparar

Este módulo compara los parámetros bidimensionales y tridimensionales de las operaciones de la pieza con los de la referencia, tratando de encontrar la operación de la referencia que se asemeja a la del modelo del usuario. Es una subfunción, por lo que no devuelve ningún valor a *Principal* o *Principal2*. Su objetivo es identificar qué operaciones son iguales en ambas piezas, independientemente de su posición en el árbol de modelado, y establecer las discrepancias entre las mismas. Por último, mostrará por el menú principal la nota que se le asigna a la pieza del usuario.

Las entradas de esta subfunción son:

- Las matrices *mat1* y *mat2*, que son resultado de la función *AnalisisOperaciones*.
- La hoja 3 del archivo Excel de resultados.
- Los *ListBox1* y *ListBox2*, asociados a la ventana de resultados y a la ventana de puntuación del menú principal, respectivamente.

- El valor de la nota actualizada por los anteriores módulos.

En primer lugar, se guarda en las variables *cont* y *contRef* el número de filas de las matrices *mat1* y *mat2*, respectivamente. [7].

```
cont = UBound(mat1, 1) - LBound(mat1, 1)
contRef = UBound(mat2, 1) - LBound(mat2, 1)
```

Código 3.51 Definición de los parámetros cont y contRef del módulo Comparar.

En el caso de que *cont* sea mayor que *contRef*, se completa la matriz *mat2* con filas vacías hasta alcanzar un número igual al de la matriz *mat1*. Esto es consecuencia del método que implementa este módulo, que altera el orden de las filas de *mat2* para que las operaciones que son semejantes en ambas piezas se almacenen en la misma fila en ambas matrices. Por tanto, si se da el caso de que *cont* es mayor que *contRef*, y no se modifica *mat2*, salta un error al no disponer la matriz de referencia de tantas filas como la matriz de la pieza del usuario.

A continuación, se establecen los pesos que tendrá cada operación en la nota de la pieza. Se declaran las siguientes variables:

- *operacionesConValores3D*: almacena el número de operaciones que de las que se pueden extraer parámetros tridimensionales. Como es un contador, en el instante inicial vale cero.
- *operacionesConValores2D*: almacena el número de operaciones que de las que se pueden extraer parámetros bidimensionales (operaciones que tienen sketch). Como es un contador, en el instante inicial vale cero.
- *pesoCadaOperacion3D*: establece, como su nombre indica, el peso que tendrá cada operación en la correción del análisis tridimensional. Caber recordar que vale un 15 % de la nota de la pieza.
- *pesoCadaOperacion2D*: establece, como su nombre indica, el peso que tendrá cada operación en la correción del análisis bidimensional. Caber recordar que vale un 15 % de la nota de la pieza.
- *nota*: equivale al porcentaje que tiene este módulo en la puntuación final de la pieza, que es un 30 %. Por tanto, se iguala a tres.

Para definirlas, se establece un bucle *For* con un número de iteraciones igual al valor almacenado por *cont*. En cada iteración, se evalúa la segunda columna de la matriz *mat1*, que es la que guarda el tipo de la operación. Se distinguen tres casos:

- Operaciones con parámetros tridimensionales y bidimensionales. Estas son:
  - *Pad*
  - *Hole*
  - *Pocket*
  - *Stiffener*
  - *Multi-Sections Solid*
  - *Shaft*
  - *Groove*

  Si la operación que se evalúa coincide con alguna de las anteriores, se aumentan los contadores *operacionesConValores3D* y *operacionesConValores2D*.

operacionesConValores3D = operacionesConValores3D + 1
operacionesConValores2D = operacionesConValores2D + 1

- Operaciones que solo tienen parámetros tridimensionales. Estas son:
  - *Mirror*
  - *Edge Fillet*
  - *Rectangular Pattern*
  - *Circular Pattern*
  - *Chamfer*
  - *Draft*
  - *Shell*

  Si la operación que se evalúa coincide con alguna de las anteriores, se aumenta el contador *operacionesConValores3D*.

  operacionesConValores3D = operacionesConValores3D + 1

- Operaciones que solo tienen parámetros bidimensionales. Estas son:
  - *Rib*
  - *Slot*

  Si la operación que se evalúa coincide con alguna de las anteriores, se aumenta el contador *operacionesConValores2D*.

  operacionesConValores2D = operacionesConValores2D + 1

Una vez establecidos los contadores, se pasan a definir las variables *pesoCadaOperacion3D* y *pesoCadaOperacion2D*, teniendo en cuenta el peso del análisis tridimensional y bidimensional en la nota de la pieza.

pesoCadaOperacion3D = 1.5 / operacionesConValores3D
pesoCadaOperacion2D = 1.5 / operacionesConValores2D

El siguiente paso que realiza el módulo es declarar una serie de variables. Entre ellas, destaca la matriz *discrepancias*, que tiene tres filas y un número de columnas igual al valor que almacena *cont*. En la primera fila se guardan las diferencias tridimensionales entre la pieza y la referencia, en la segunda fila las diferencias bidimensionales, y en la tercera fila la suma de las dos. También se definen una serie de matrices que almacenan el número y las magnitudes de los elementos geométricos de los *sketches* de cada operación.

A continuación, se realiza un bucle para relacionar aquellas operaciones que son semejantes en ambas piezas. El método empleado se recoge en el diagrama de flujo de la *Figura 3.14*.

En el diagrama aparece la variable *pos*, que indica cuántas operaciones del mismo tipo hay en la referencia. La matriz *discrepancias* se completa para un mismo tipo de operación y luego se borra para completarse con el siguiente. Por tanto, *pos* me indica el número de columnas con información de la matriz. Por otro lado, se nombra que se reordena la matriz de referencia (*mat2*). Esto se lleva a cabo para que en la misma fila de *mat1* y *mat2* las operaciones sean semejantes.

Una vez se tiene ordenada la matriz de referencia, CPIT vuelve a compara las matrices fila por fila para indicarle al usuario los errores que ha cometido a través de la ventana de resultados. Por ejemplo, si la altura de un *pad* es errónea, se escribe el siguiente código:

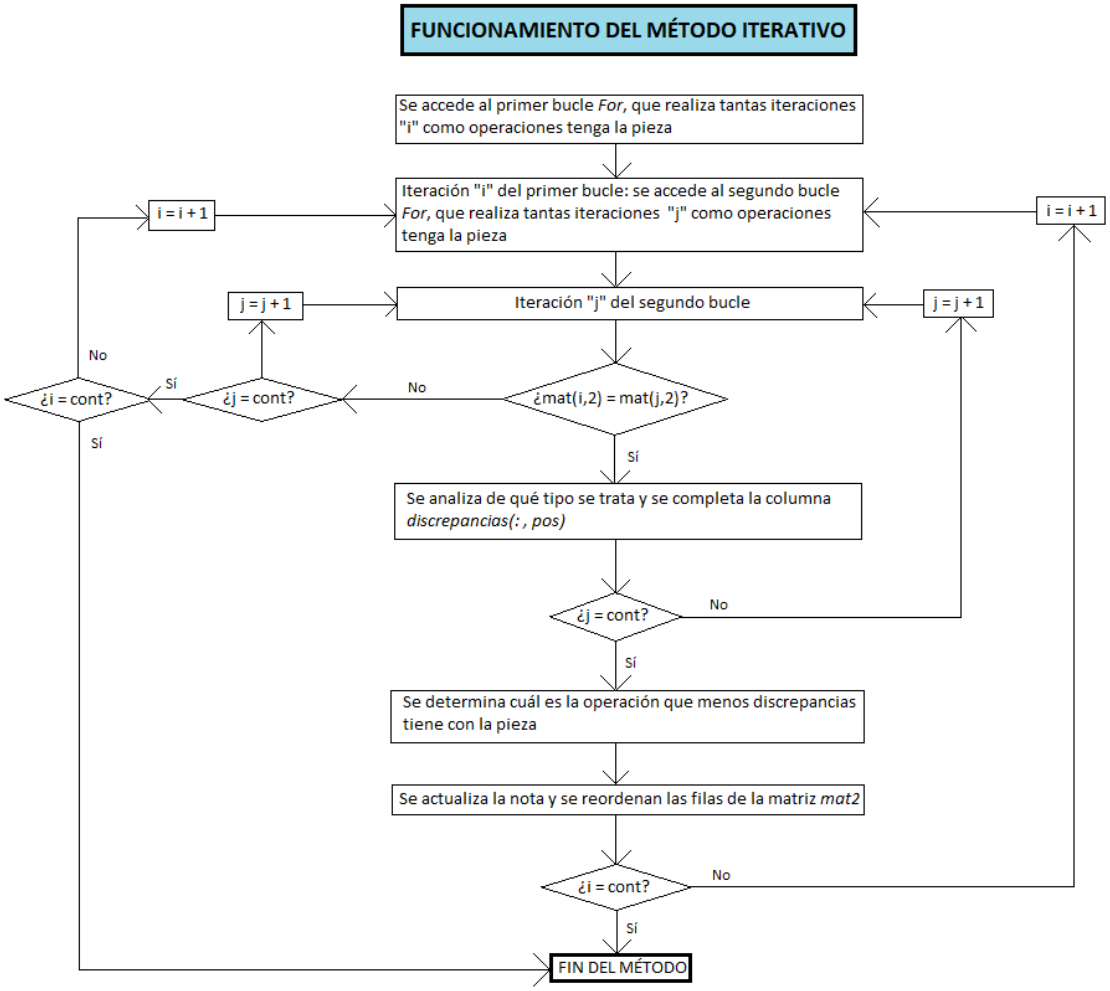


Figura 3.14 Diagrama de flujo sobre cómo CPIT compara las operaciones.

```
ListBox1.AddItem "Corrige la altura del" & " " & mat1(i, 1)
```

Código 3.52 Mostrar consejos de corrección a través del menú principal.

Para finalizar, este módulo escribe en el menú principal la puntuación que estima que tiene la pieza, en base a los errores que se han ido detectando durante su ejecución. La nota aparecerá redondeada a una cifra significativa, para lo que se ha empleado el código:

```
ListBox2.AddItem Round(nota + notaAct, 1)
```

Código 3.53 Mostrar la nota final a través del menú principal.

Además, la ventana se sombreará de verde si la pieza está aprobada (nota ≥ 5), mientras que si está supensa (nota < 5) se sombreará de rojo.

4 Análisis y discusión de resultados

En este capítulo se va a emplear CPIT para comparar una serie de piezas con las referencias introducidas en el *Capítulo 2*. El objetivo es comprobar si la aplicación es capaz de detectar los fallos cometidos por el usuario en su modelo, y examinar los resultados obtenidos tras el análisis.

4.1 Análisis de la pieza 1

Suponga que se le ha aportado a un alumno el plano con las cotas, *Figura 2.2*, de la pieza mostrada en la *Figura 2.1*. Sin embargo, a la hora de modelarla en CATIA, comete los siguientes errores:

- El *pad* es cinco milímetros inferior al de la referencia. Consecuentemente, el *pocket* (agujero de mayor diámetro) tiene menos profundidad, y la longitud y la coordenada z del *hole* de la cara superior difiere a la de la referencia.
- El tipo de los *hole* es distinto. Mientras que en la referencia son *counterdrilled*, en la pieza del usuario son *simple*.

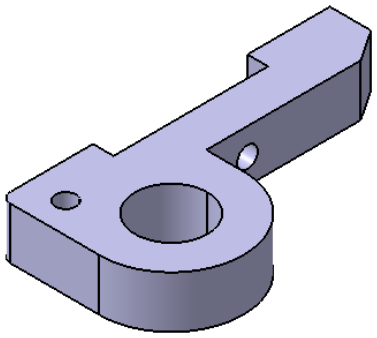


Figura 4.1 Pieza 1 realizada por el usuario.

En primer lugar, el alumno debe abrir CPIT, buscar el directorio de la referencia, y seleccionar cómo quiere realizar el análisis. En este caso, marca que sí quiere ver los resultados exportados a

Excel y que dispone del archivo de extensión *.CATPart* de referencia.

Una vez CPIT analiza la pieza, el menú queda como el de la imagen.

Figura 4.2 Menú tras analizar la pieza 1.

Como se puede ver, la aplicación le da un 4,8 de puntuación a la pieza. Para entender mejor cómo ha valorado CPIT la pieza, se adjunta la siguiente hoja de Excel donde aparece el peso de cada operación en la nota final y la penalización de cada error cometido.

NIVEL 1 (40% de la nota)				
Número de parámetros	Peso de cada parámetro		Errores cometidos	Puntuación NIVEL 1
10	Bounding Box (3 coordenadas)	3/1,8 = 0,6	10	4 - (1,2 + 1 + 1,8) = 0
	Volumen	1		
	COG y Momentos de Inercia (6 coordenadas)	6/1,2 = 0,2		

NIVEL 2 (45% de la nota) ----------> Número de operaciones (20%) , orden del árbol de modelado (10%) y análisis 3D (15%)				
ÁRBOL DE MODELADO				
¿Coinciden las operaciones de la pieza y de la referencia?			Puntuación número de operaciones	
SÍ			2	
¿Coincide el orden del árbol de modelado?			Puntuación orden del árbol de modelado	
No, dos operaciones están mal situadas			0,6	
ANÁLISIS 3D DE LAS OPERACIONES				
Número de operaciones	Peso	Errores cometidos	Peso de cada error	Puntuación análisis 3D
5	1,5/5 = 0,3	4	*Hole.1:* 0,3/6	0,8
			Hole.3: 0,3/6	
			Pad.1: 0,3	
			Pocket.1: 0,3	

NIVEL 3 (30% de la nota)				
Número de operaciones	Peso	Errores cometidos	Peso de cada error	Puntuación NIVEL 3
4	1,5/4 = 0,375	1	*Hole.1:* 0,375x0,3	1,3875

NOTA FINAL DE LA PIEZA
4,8

Figura 4.3 Cálculo de la nota de la pieza 1.

La primera hoja del archivo Excel de resultados, asociada al nivel 1 de análisis, muestra que todos los parámetros generales de la pieza del usuario difieren de los de la referencia.

PARÁMETROS GENERALES

Parámetros	Referencia	Pieza	Discrepancias	Comentarios
BBLx (mm)	167,7521027	167,6849901	Incorrecto	La coordenada x del Bounding Box de su pieza se encuentra más retrasada que la de la referencia
BBLy (mm)	88,88986805	88,74287572	Incorrecto	La coordenada y del Bounding Box de su pieza se encuentra más retrasada que la de la referencia
BBLz (mm)	30,33014752	25	Incorrecto	La coordenada z del Bounding Box de su pieza se encuentra por debajo de la de la referencia
Volumen (m3)	0,000180447	0,000151259	Incorrecto	El volumen de su pieza es inferior al de la referencia
COGx (m)	0,096824281	0,096952918	Incorrecto	La coordenada x del centro de gravedad de su pieza se encuentra más adelantada que la de la referencia
COGy (m)	0,031098404	0,030998052	Incorrecto	La coordenada y del centro de gravedad de su pieza se encuentra más retrasada que la de la referencia
COGz (m)	0,014994106	0,012505192	Incorrecto	La coordenada z del centro de gravedad de su pieza se encuentra por debajo de la de la referencia
M1 (kg*m2)	8,33199E-05	6,6453E-05	Incorrecto	El momento de su pieza es menor que de la referencia
M2 (kg*m2)	0,0004157	0,000344452	Incorrecto	El momento de su pieza es menor que de la referencia
M3 (kg*m2)	0,000471785	0,000395006	Incorrecto	El momento de su pieza es menor que de la referencia

Figura 4.4 Hoja 1 del análisis de la pieza 1.

La segunda hoja del archivo Excel de resultados, asociada al nivel dos de análisis, muestra que la pieza y la referencia tienen el mismo número de operaciones. Además, muestra los valores tridimensionales de cada una de ellas, donde se pueden observar qué operaciones son distintas en ambas piezas.

PIEZA

Orden modelado	
Pad.1	(Pad)
EdgeFillet.1	(ConstRadEdgeFillet)
Hole.1	(Hole)
Pocket.1	(Pocket)
Hole.3	(Hole)

Tipo de operación	Cantidad
Pad	1
ConstRadEdgeFillet	1
Hole	2
Pocket	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	4

Figura 4.5 Hoja 2 (árbol de modelado) pieza 1.

REFERENCIA

Orden modelado	
Pad.1	(Pad)
Pocket.1	(Pocket)
Hole.1	(Hole)
EdgeFillet.1	(ConstRadEdgeFillet)
Hole.2	(Hole)

Tipo de operación	Cantidad
Pad	1
Pocket	1
Hole	2
ConstRadEdgeFillet	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	4

Figura 4.6 Hoja 2 (árbol de modelado) referencia 1.

Dado que en la hoja dos del Excel el ancho de las columnas es muy elevado para que todos los datos queden confinados en sus respectivas tablas, se ha optado por poner los resultados obtenidos para la pieza y para la referencia por separado (*Figura 4.5* y *Figura 4.6* respectivamente).

A continuación se muestra cómo queda la segunda parte de la segunda hoja del archivo Excel de resultados. Al igual que se hizo para los resultados del árbol de modelado, se han separado los de la pieza y los de la referencia (*Figura 4.7* y *Figura 4.8* respectivamente).

PIEZA

Pad

Nombre	Altura (mm)
Pad.1	25

ConstRadEdgeFillet

Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)
EdgeFillet.1	5	1	25

Hole

Nombre	Tipo	Diámetro (mm)	Profundidad (mm)	Extensión	Diámetro Superior (mm)	Profundidad Superior (mm)	Ángulo (º)
Hole.1	0	10	25	2	-	-	-
Hole.3	0	10	20	2	-	-	-

Pocket

Nombre	Profundidad (mm)
Pocket.1	25

Figura 4.7 Hoja 2 (análisis 3D) pieza 1.

REFERENCIA

Pad	
Nombre	Altura (mm)
Pad.1	30

Pocket	
Nombre	Profundidad (mm)
Pocket.1	30

Hole							
Nombre	Tipo	Diámetro (mm)	Profundidad (mm)	Extensión	Diámetro Superior (mm)	Profundidad Superior (mm)	Ángulo (º)
Hole.1	4	10	30	2	15	5	60
Hole.2	4	10	20	2	15	5	60

ConstRadEdgeFillet			
Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)
EdgeFillet.1	5	1	30

Figura 4.8 Hoja 2 (análisis 3D) referencia 1.

Por último, la tercera hoja del archivo Excel de resultados, asociada al nivel tres de análisis, muestra las dimensiones de los elementos que forman los *sketches* de cada operación.

PIEZA

Pad.1	(Sketch.1)

Nº líneas	Nº puntos	Nº circunferencias
10	12	1

Nombre	Longitud (mm)
Line.1	40
Line.2	10
Line.3	75
Line.4	10
Line.5	45
Line.6	48
Line.7	18
Line.8	75
Line.9	18,02775638
Line.10	20

Nombre	Coord Horz (mm)	Coord Vert (mm)
Point.2	40	0
Point.3	40	10
Point.4	115	10
Point.5	115	0
Point.6	160	0
Point.7	160	48
Point.8	125	48
Point.9	90	48
Point.10	90	30
Point.11	15	30
Point.12	0	20
Point.13	0	0

Nombre	Radio (mm)
Circle.1	35

Figura 4.9 Hoja 3 del análisis de la pieza 1.

Se han adjuntado solo en la *Figura 4.9* y en la *Figura 4.10* los valores correspondientes a la operación *Pad.1*. Las tablas del resto de operaciones serían análogas, variando el número de elementos geométricos que conforman sus *sketches*.

Una vez observados los valores que CPIT extrae de ambas piezas, se puede pasar a examinar la información mostrada en la ventana de resultados del menú principal. Como se puede observar en la *Figura 4.2*, CPIT informa a través de la ventana que los módulos se están ejecutando correctamente. Como el número de operaciones de la pieza y de la referencia coincide, y existen fallos en los parámetros generales, la aplicación sigue ejecutándose para encontrar dónde se han cometido los mismos. Una vez compara las operaciones de ambas piezas, muestra una serie de consejos para que el usuario pueda corregir su modelo. *Figura 4.11*.

REFERENCIA	
Pad.1	(Sketch.1)

Nº líneas	Nº puntos	Nº circunferencias
10	12	1

Nombre	Longitud (mm)
Line.1	40
Line.2	10
Line.3	75
Line.4	10
Line.5	45
Line.6	48
Line.7	18
Line.8	75
Line.9	18,02775638
Line.10	20

Nombre	Coord Horz (mm)	Coord Vert (mm)
Point.1	1,42109E-14	-3,34449E-14
Point.3	40	-6,67652E-14
Point.4	40	10
Point.5	115	10
Point.6	115	-1,29241E-13
Point.7	160	-1,66726E-13
Point.8	160	48
Point.9	125	48
Point.10	90	48
Point.11	90	30
Point.12	15	30
Point.13	2,84217E-14	20

Nombre	Radio (mm)
Circle.1	35

Figura 4.10 Hoja 3 del análisis de la referencia 1.

- CORRECCIONES A REALIZAR:
Corrige la altura del Pad.1
El tipo del Hole.1 no coincide con el de referencia
Corrige la profundidad del Pocket.1
El tipo del Hole.3 no coincide con el de referencia

Figura 4.11 Consejos de CPIT para corregir la pieza 1.

En conclusión, CPIT ha analizado correctamente la pieza modelada por el alumno, dándole una calificación acorde al resultado obtenido. Además, los consejos mostrados a través de la ventana de resultados concuerdan con los fallos que tiene la pieza.

4.2 Análisis de la pieza 2

Suponga ahora que se le ha aportado a un alumno el plano con las cotas, *Figura 2.4*, de la pieza mostrada en la *Figura 2.3*. Sin embargo, a la hora de modelarla en CATIA, comete los siguientes errores:

- El número de patrones del *hole* de la corona de la tubería son seis, mientras que en la referencia son ocho.
- Los refuerzos son de un milímetro y medio de espesor en lugar de un milímetro.

Al igual que en la pieza anterior, lo primero que el usuario debe hacer es abrir CPIT, buscar el directorio de la referencia, y seleccionar cómo quiere hacer el análisis. Se escogen las mismas opciones de análisis que en la pieza 1. Una vez CPIT analiza la pieza, el menú queda como el de la *Figura 4.13*.

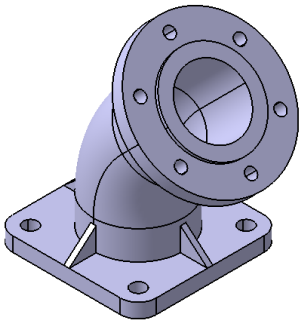


Figura 4.12 Pieza 2 realizada por el usuario.

Figura 4.13 Menú tras analizar la pieza 2.

Como se puede ver, la aplicación le da un 6,6 de puntuación a la pieza. Para entender mejor cómo ha valorado CPIT la pieza, se adjunta la siguiente hoja de Excel donde aparece el peso de cada operación en la nota final y la penalización de cada error cometido.

NIVEL 1 (40% de la nota)				
Número de parámetros	Peso de cada parámetro		Errores cometidos	Puntuación NIVEL 1
10	Bounding Box (3 coordenadas)	3/1,8 = 0,6	8	4 - (1 + 0,2x5+0,6x2) = 2
	Volumen	1		
	COG y Momentos de Inercia (6 coordenadas)	6/1,2 = 0,2		

NIVEL 2 (45% de la nota) ----------> Número de operaciones (20%) , orden del árbol de modelado (10%) y análisis 3D (15%)				
ÁRBOL DE MODELADO				
¿Coinciden las operaciones de la pieza y de la referencia?			Puntuación número de operaciones	
Sí			2	
¿Coincide el orden del árbol de modelado?			Puntuación orden del árbol de modelado	
Sí			1	
ANÁLISIS 3D DE LAS OPERACIONES				
Número de operaciones	Peso	Errores cometidos	Peso de cada error	Puntuación análisis 3D
13	1,5/13 = 0,1154	2	*CircPattern.1:* 0,1154/2 *Stiffener.1:* 0,1154	1,3269

NOTA FINAL DE LA PIEZA
6,6

NIVEL 3 (30% de la nota)				
Número de operaciones	Peso	Errores cometidos	Peso de cada error	Puntuación NIVEL 3
7	1,5/7 = 0,2143	0	-	1,5

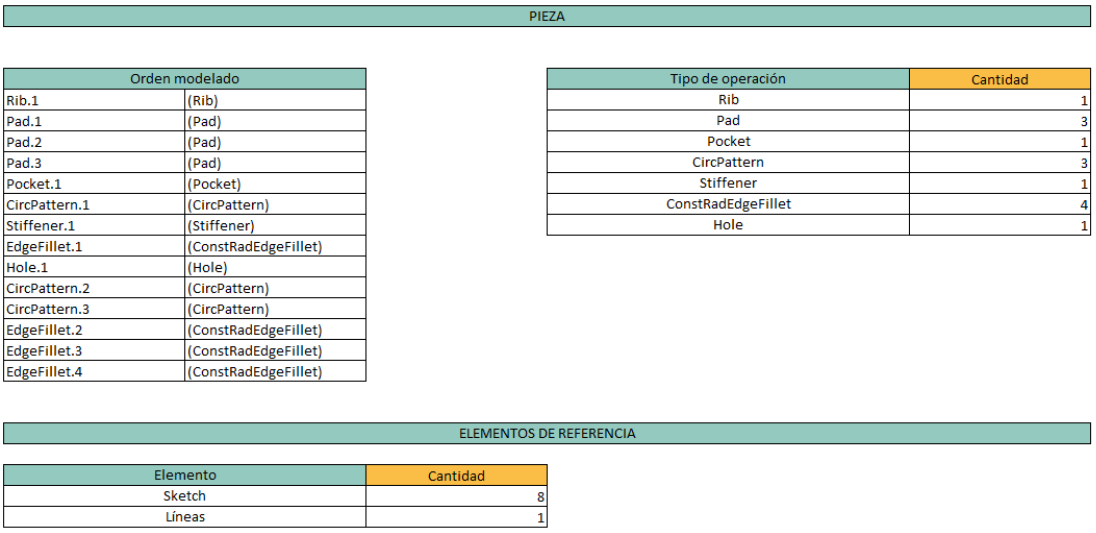
Figura 4.14 Cálculo de la nota de la pieza 2.

PARÁMETROS GENERALES

Parámetros	Referencia	Pieza	Discrepancias	Comentarios
BBLx (mm)	63,13337377	63,13431669	Incorrecto	La coordenada x del Bounding Box de su pieza se encuentra más adelantada que la de la referencia
BBLy (mm)	40	40	Correcto	Esta coordenada coincide en su pieza y en la de referencia.
BBLz (mm)	39,45450452	39,43758929	Incorrecto	La coordenada z del Bounding Box de su pieza se encuentra por debajo de la de la referencia
Volumen (m3)	1,98401E-05	1,97346E-05	Incorrecto	El volumen de su pieza es inferior al de la referencia
COGx (m)	4,93209E-12	2,28777E-11	Correcto	Esta coordenada coincide en su pieza y en la de referencia.
COGy (m)	0,006019711	0,00599842	Incorrecto	La coordenada y del centro de gravedad de su pieza se encuentra más retrasada que la de la referencia
COGz (m)	0,02148703	0,021466096	Incorrecto	La coordenada z del centro de gravedad de su pieza se encuentra por debajo de la de la referencia
M1 (kg*m2)	3,7161E-06	3,69182E-06	Incorrecto	El momento de su pieza es menor que de la referencia
M2 (kg*m2)	8,29645E-06	8,23717E-06	Incorrecto	El momento de su pieza es menor que de la referencia
M3 (kg*m2)	8,33932E-06	8,2786E-06	Incorrecto	El momento de su pieza es menor que de la referencia

Figura 4.15 Hoja 1 del análisis de la pieza 2.

La primera hoja del archivo Excel de resultados, asociada al nivel 1 de análisis, muestra que todos los parámetros generales de la pieza del usuario, salvo dos, difieren de los de la referencia. La segunda hoja del archivo Excel de resultados, muestra el árbol de modelado de la pieza y de la referencia, así como el análisis 3D de sus operaciones.



PIEZA

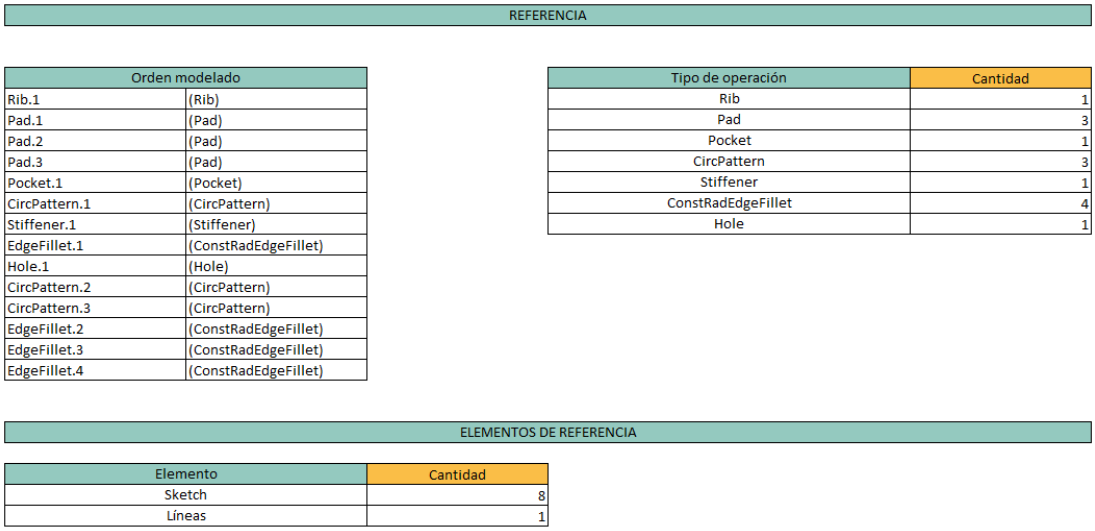
Orden modelado	
Rib.1	(Rib)
Pad.1	(Pad)
Pad.2	(Pad)
Pad.3	(Pad)
Pocket.1	(Pocket)
CircPattern.1	(CircPattern)
Stiffener.1	(Stiffener)
EdgeFillet.1	(ConstRadEdgeFillet)
Hole.1	(Hole)
CircPattern.2	(CircPattern)
CircPattern.3	(CircPattern)
EdgeFillet.2	(ConstRadEdgeFillet)
EdgeFillet.3	(ConstRadEdgeFillet)
EdgeFillet.4	(ConstRadEdgeFillet)

Tipo de operación	Cantidad
Rib	1
Pad	3
Pocket	1
CircPattern	3
Stiffener	1
ConstRadEdgeFillet	4
Hole	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	8
Líneas	1

Figura 4.16 Hoja 2 (árbol de modelado) pieza 2.



REFERENCIA

Orden modelado	
Rib.1	(Rib)
Pad.1	(Pad)
Pad.2	(Pad)
Pad.3	(Pad)
Pocket.1	(Pocket)
CircPattern.1	(CircPattern)
Stiffener.1	(Stiffener)
EdgeFillet.1	(ConstRadEdgeFillet)
Hole.1	(Hole)
CircPattern.2	(CircPattern)
CircPattern.3	(CircPattern)
EdgeFillet.2	(ConstRadEdgeFillet)
EdgeFillet.3	(ConstRadEdgeFillet)
EdgeFillet.4	(ConstRadEdgeFillet)

Tipo de operación	Cantidad
Rib	1
Pad	3
Pocket	1
CircPattern	3
Stiffener	1
ConstRadEdgeFillet	4
Hole	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	8
Líneas	1

Figura 4.17 Hoja 2 (árbol de modelado) referencia 2.

Al igual que sucedió en la *Pieza 1*, el ancho de las columnas es muy elevado para que todos los datos queden confinados en sus respectivas tablas, se ha optado por poner los resultados obtenidos para la pieza y para la referencia por separado (*Figura 4.16* y *Figura 4.17* respectivamente).

A continuación se muestran los resultados obtenidos del análisis tridimensional de las operaciones de la pieza y de la referencia. Se han adjuntado los valores tridimensionales de la pieza (*Figura 4.18*) y de la referencia (*Figura 4.19*) por separado.

PIEZA

Rib
Nombre
Rib.1

Pad	
Nombre	Altura (mm)
Rib.1	
Pad.2	1
Pad.3	4

Pocket	
Nombre	Profundidad (mm)
Pocket.1	1

CircPattern		
Nombre	Número	Pieza repetida
CircPattern.1	8	Pocket.1
CircPattern.2	4	Stiffener.1
CircPattern.3	4	Hole.1

Stiffener	
Nombre	Espesor (mm)
Stiffener.1	1

ConstRadEdgeFillet			
Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)
EdgeFillet.1	5	1	4
EdgeFillet.2	5	1	4
EdgeFillet.3	5	1	4
EdgeFillet.4	5	1	4

Hole							
Nombre	Tipo	Diámetro (mm)	Profundidad (mm)	Extensión	Diámetro Superior (mm)	Profundidad Superior (mm	Ángulo (º)
Hole.1	0	4	4	2	-	-	-

Figura 4.18 Hoja 2 (análisis 3D) pieza 2.

REFERENCIA

Rib
Nombre
Rib.1

Pad	
Nombre	Altura (mm)
Rib.1	
Pad.2	1
Pad.3	4

Pocket	
Nombre	Profundidad (mm)
Pocket.1	1

CircPattern		
Nombre	Número	Pieza repetida
CircPattern.1	6	Pocket.1
CircPattern.2	4	Stiffener.1
CircPattern.3	4	Hole.1

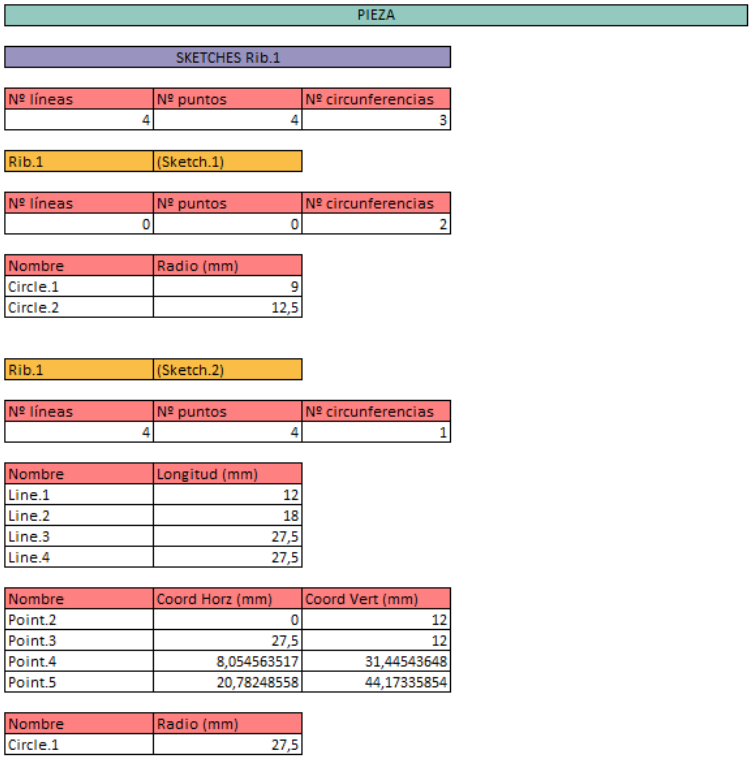
Stiffener	
Nombre	Espesor (mm)
Stiffener.1	1,5

ConstRadEdgeFillet			
Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)
EdgeFillet.1	5	1	4
EdgeFillet.2	5	1	4
EdgeFillet.3	5	1	4
EdgeFillet.4	5	1	4

Hole							
Nombre	Tipo	Diámetro (mm)	Profundidad (mm)	Extensión	Diámetro Superior (mm)	Profundidad Superior (mm	Ángulo (º)
Hole.1	0	4	4	2	-	-	-

Figura 4.19 Hoja 2 (análisis 3D) referencia 2.

Por último, la tercera hoja del archivo Excel de resultados, asociada al nivel tres de análisis, muestra las dimensiones de los elementos que forman los *sketches* de cada operación. En la *Figura 4.20* y en la *Figura 4.21*, se recogen los dos *sketches* del primer *rib* de la pieza como de la referencia, respectivamente.



PIEZA

SKETCHES Rib.1

Nº líneas	Nº puntos	Nº circunferencias
4	4	3

Rib.1	(Sketch.1)

Nº líneas	Nº puntos	Nº circunferencias
0	0	2

Nombre	Radio (mm)
Circle.1	9
Circle.2	12,5

Rib.1	(Sketch.2)

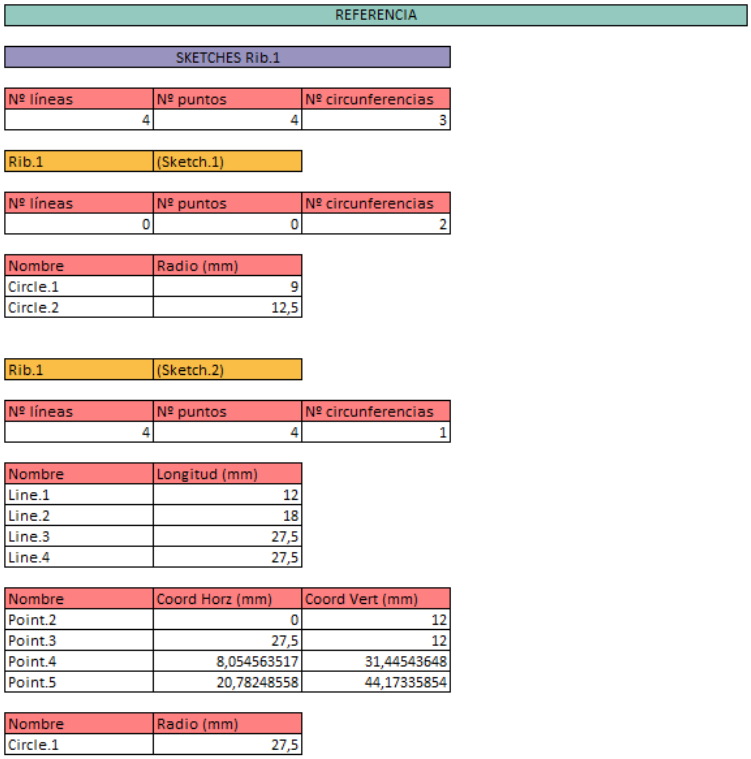
Nº líneas	Nº puntos	Nº circunferencias
4	4	1

Nombre	Longitud (mm)
Line.1	12
Line.2	18
Line.3	27,5
Line.4	27,5

Nombre	Coord Horz (mm)	Coord Vert (mm)
Point.2	0	12
Point.3	27,5	12
Point.4	8,054563517	31,44543648
Point.5	20,78248558	44,17335854

Nombre	Radio (mm)
Circle.1	27,5

Figura 4.20 Hoja 3 del análisis de la pieza 2.



REFERENCIA

SKETCHES Rib.1

Nº líneas	Nº puntos	Nº circunferencias
4	4	3

Rib.1	(Sketch.1)

Nº líneas	Nº puntos	Nº circunferencias
0	0	2

Nombre	Radio (mm)
Circle.1	9
Circle.2	12,5

Rib.1	(Sketch.2)

Nº líneas	Nº puntos	Nº circunferencias
4	4	1

Nombre	Longitud (mm)
Line.1	12
Line.2	18
Line.3	27,5
Line.4	27,5

Nombre	Coord Horz (mm)	Coord Vert (mm)
Point.2	0	12
Point.3	27,5	12
Point.4	8,054563517	31,44543648
Point.5	20,78248558	44,17335854

Nombre	Radio (mm)
Circle.1	27,5

Figura 4.21 Hoja 3 del análisis de la referencia 2.

La *Figura 4.20* y la *Figura 4.21* no están completas, ya que en la hoja aparecen los *sketches* de cada una de las operaciones que conforman ambas piezas.

Una vez observados los valores que CPIT extrae de ambas piezas, se puede pasar a examinar la información mostrada en la ventana de resultados del menú principal. Como se puede observar en la *Figura 4.13*, CPIT informa a través de la ventana que los módulos se están ejecutando correctamente. Como el número de operaciones de la pieza y de la referencia coincide, y existen fallos en los parámetros generales, la aplicación sigue ejecutándose para encontrar dónde se han cometido los mismos. Una vez compara las operaciones de ambas piezas, muestra una serie de consejos para que el usuario pueda corregir su modelo.

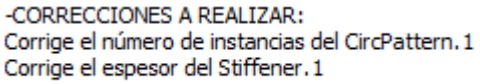


Figura 4.22 Consejos de CPIT para corregir la pieza 2.

En conclusión, CPIT ha analizado correctamente la pieza modelada por el alumno, dándole una calificación acorde al resultado obtenido. Además, los consejos mostrados a través de la ventana de resultados concuerdan con los fallos que tiene la pieza.

4.3 Análisis de la pieza 3

Suponga que se le ha aportado a un alumno el plano con las cotas, *Figura 2.6*, de la pieza mostrada en la *Figura 2.5*. Sin embargo, a la hora de modelarla en CATIA, comete los siguientes errores:

- El respaldo es dos milímetros más fino que el de la referencia. Además, los respaldos tienen distinta forma, por lo que el análisis bidimensional será fundamental para encontrar las discrepancias existentes en ambos modelos.
- Las baldas son un milímetro más estrechas que las de la referencia.
- Los *chamfer* realizados tienen distinta longitud y ángulo que los de la referencia.

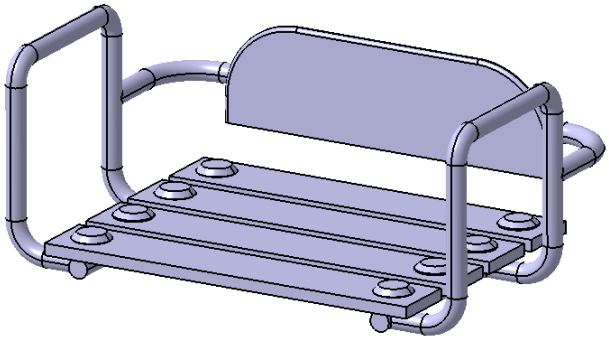


Figura 4.23 Pieza 3 realizada por el usuario.

El grado de dificultad de esta pieza es clave para comprobar la efectividad de CPIT. Una vez analizada esta pieza, se dará por concluida la prueba del programa, pasando a realizar una serie de conclusiones en el *Capítulo 5* sobre los resultados obtenidos .

Al igual que en las piezas anteriores, lo primero que el usuario debe hacer es abrir CPIT, buscar el directorio de la referencia, y seleccionar cómo quiere hacer el análisis. Se escogen las mismas opciones de análisis que en la pieza 1.

Figura 4.24 Menú tras analizar la pieza 3.

Como se puede ver, la aplicación le da un 5,4 de puntuación a la pieza. Para entender mejor cómo ha valorado CPIT la pieza, se adjunta la siguiente hoja de Excel donde aparece el peso de cada operación en la nota final y la penalización de cada error cometido.

NIVEL 1 (30% de la nota)				
Número de parámetros	Peso de cada parámetro		Errores cometidos	Puntuación NIVEL 1
10	Bounding Box (3 coordenadas)	3/1,8 = 0,6	10	4 - (1,8 + 1 + 1,2) = 0
	Volumen	1		
	COG y Momentos de Inercia (6 coordenadas)	6/1,2 = 0,2		

NIVEL 2 (45% de la nota) ----------> Número de operaciones (20%) , orden del árbol de modelado (10%) y análisis 3D (15%)				
ÁRBOL DE MODELADO				
¿Coinciden las operaciones de la pieza y de la referencia?			Puntuación número de operaciones	
Sí			2	
¿Coincide el orden del árbol de modelado?			Puntuación orden del árbol de modelado	
Sí			1	
ANÁLISIS 3D DE LAS OPERACIONES				
Número de operaciones	Peso	Errores cometidos	Peso de cada error	Puntuación análisis 3D
10	1,5/10 = 0,15	6	*Pad.3:* 0,15	1
			Pad.5: 0,15	
			Chamfer.1: (0,15/3)x2	
			Chamfer.2: (0,15/3)x2	

NOTA FINAL DE LA PIEZA
5,4

NIVEL 3 (30% de la nota)				
Número de operaciones	Peso	Errores cometidos	Peso de cada error	Puntuación NIVEL 3
8	1,5/8 = 0,1875	*Sketch.17*	*Pad.5:* (0,1875/14)*11	1,35267
		Puntos: 6		
		Líneas: 3		
		Circunferencias: 2		

Figura 4.25 Cálculo de la nota de la pieza 3.

La primera hoja del archivo Excel de resultados, asociada al nivel 1 de análisis, muestra que todos los parámetros generales de la pieza del usuario difieren de los de la referencia.

PARÁMETROS GENERALES

Parámetros	Referencia	Pieza	Discrepancias	Comentarios
BBLx (mm)	380,2000122	380,2011348	Incorrecto	La coordenada x del Bounding Box de su pieza se encuentra más adelantada que la de la referencia
BBLy (mm)	260,7927434	258,5473716	Incorrecto	La coordenada y del Bounding Box de su pieza se encuentra más retrasada que la de la referencia
BBLz (mm)	191,2673696	179,5042292	Incorrecto	La coordenada z del Bounding Box de su pieza se encuentra por debajo de la de la referencia
Volumen (m3)	0,001143343	0,001016385	Incorrecto	El volumen de su pieza es inferior al de la referencia
COGx (m)	2,47504E-12	3,76721E-06	Incorrecto	La coordenada x del centro de gravedad de su pieza se encuentra más adelantada que la de la referencia
COGy (m)	-0,06708857	-0,072882301	Incorrecto	La coordenada y del centro de gravedad de su pieza se encuentra más retrasada que la de la referencia
COGz (m)	0,037159531	0,033306139	Incorrecto	La coordenada z del centro de gravedad de su pieza se encuentra por debajo de la de la referencia
M1 (kg*m2)	0,009169122	0,007761217	Incorrecto	El momento de su pieza es menor que de la referencia
M2 (kg*m2)	0,015766503	0,014660318	Incorrecto	El momento de su pieza es menor que de la referencia
M3 (kg*m2)	0,021209783	0,01924155	Incorrecto	El momento de su pieza es menor que de la referencia

Figura 4.26 Hoja 1 del análisis de la pieza 3.

La segunda hoja del archivo Excel de resultados, asociada al nivel dos de análisis, muestra que la pieza y la referencia tienen el mismo número de operaciones. Además, muestra los valores tridimensionales de cada una de ellas, donde se pueden observar qué operaciones son distintas en ambas piezas.

PIEZA

Orden modelado	
Rib.1	(Rib)
Pad.1	(Pad)
Rib.2	(Rib)
Rib.3	(Rib)
Mirror.1	(Mirror)
Rib.6	(Rib)
Pad.3	(Pad)
RectPattern.1	(RectPattern)
Pad.4	(Pad)
RectPattern.2	(RectPattern)
Chamfer.1	(Chamfer)
Chamfer.2	(Chamfer)
Pad.5	(Pad)
EdgeFillet.1	(ConstRadEdgeFillet)

Tipo de operación	Cantidad
Rib	4
Pad	4
Mirror	1
RectPattern	2
Chamfer	2
ConstRadEdgeFillet	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	14
Planos	12

Figura 4.27 Hoja 2 (árbol de modelado) pieza 3.

REFERENCIA

Orden modelado	
Rib.1	(Rib)
Pad.1	(Pad)
Rib.2	(Rib)
Rib.3	(Rib)
Mirror.1	(Mirror)
Rib.6	(Rib)
Pad.3	(Pad)
RectPattern.1	(RectPattern)
Pad.4	(Pad)
RectPattern.2	(RectPattern)
Chamfer.1	(Chamfer)
Chamfer.2	(Chamfer)
Pad.5	(Pad)
EdgeFillet.1	(ConstRadEdgeFillet)

Tipo de operación	Cantidad
Rib	4
Pad	4
Mirror	1
RectPattern	2
Chamfer	2
ConstRadEdgeFillet	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	14
Planos	12

Figura 4.28 Hoja 2 (árbol de modelado) referencia 3.

Al igual que sucedió en la *Pieza 1* y en la *Pieza 2*, el ancho de las columnas es muy elevado para que todos los datos queden confinados en sus respectivas tablas, por lo que se ha optado por poner los resultados obtenidos para la pieza y para la referencia por separado (*Figura 4.27* y *Figura 4.28* respectivamente).

A continuación se muestran los resultados obtenidos del análisis tridimensional de las operaciones de la pieza y de la referencia. Se han adjuntado los valores tridimensionales de la pieza (*Figura 4.29*) y de la referencia (*Figura 4.30*) por separado.

PIEZA

Rib
Nombre
Rib.1
Rib.2
Rib.3
Rib.6

Pad	
Nombre	Altura (mm)
Rib.1	
Pad.3	9
Pad.4	5
Pad.5	4

Mirror								
Nombre	Objeto	Plano	Coord x primer VD	Coord y primer VD	Coord z primer VD	Coord x segundo VD	Coord y segundo VD	Coord z segundo VD
Mirror.1	Rib.1	Plano yz	0	1	0	0	0	1

RectPattern		
Nombre	Número	Pieza repetida
RectPattern.1	4	Pad.3
RectPattern.2	4	Pad.4

Chamfer			
Nombre	Ángulo (º)	Longitud (mm)	Elementos afectados
Chamfer.1	50	4	1
Chamfer.2	50	4	7

ConstRadEdgeFillet							
Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)	Longitud 2 (mm)	Longitud 3 (mm)	Longitud 4 (mm)	Longitud 5 (mm)
EdgeFillet.1	3	5	150	376,9911184	20	376,9911184	20

Figura 4.29 Hoja 2 (análisis 3D) pieza 3.

REFERENCIA

Rib
Nombre
Rib.1
Rib.2
Rib.3
Rib.6

Pad	
Nombre	Altura (mm)
Rib.1	
Pad.3	9,9
Pad.4	5
Pad.5	6

Mirror								
Nombre	Objeto	Plano	Coord x primer VD	Coord y primer VD	Coord z primer VD	Coord x segundo VD	Coord y segundo VD	Coord z segundo VD
Mirror.1	Rib.1	Plano yz	0	1	0	0	0	1

RectPattern		
Nombre	Número	Pieza repetida
RectPattern.1	4	Pad.3
RectPattern.2	4	Pad.4

Chamfer			
Nombre	Ángulo (º)	Longitud (mm)	Elementos afectados
Chamfer.1	45	5	1
Chamfer.2	45	5	7

ConstRadEdgeFillet							
Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)	Longitud 2 (mm)	Longitud 3 (mm)	Longitud 4 (mm)	Longitud 5 (mm)
EdgeFillet.1	3	5	180	282,7433388	54	282,7433388	54

Figura 4.30 Hoja 2 (análisis 3D) referencia 3.

Por último, la tercera hoja del archivo Excel de resultados, asociada al nivel tres de análisis, muestra las dimensiones de los elementos que forman los *sketches* de cada operación. En la *Figura 4.31* y en la *Figura 4.32* se recogen el primer *sketch* y parte del segundo del primer *rib* de la pieza y de la referencia, respectivamente.

PIEZA		
SKETCHES Rib.1		
Nº líneas	Nº puntos	Nº circunferencias
3	10	3
Rib.1	(Sketch.2)	
Nº líneas	Nº puntos	Nº circunferencias
0	1	1
Nombre	Coord Horz (mm)	Coord Vert (mm)
Proyección.1	-125	-200
Nombre	Radio (mm)	
Circle.1	7,6	
Rib.1	(Sketch.1)	
Nº líneas	Nº puntos	Nº circunferencias
3	9	2
Nombre	Longitud (mm)	
Line.1	175	
Line.2	200	
Line.3	175	

Figura 4.31 Hoja 3 del análisis de la pieza 3.

REFERENCIA		
SKETCHES Rib.1		
Nº líneas	Nº puntos	Nº circunferencias
3	10	3
Rib.1	(Sketch.2)	
Nº líneas	Nº puntos	Nº circunferencias
0	1	1
Nombre	Coord Horz (mm)	Coord Vert (mm)
Proyección.1	-125	-200
Nombre	Radio (mm)	
Circle.1	7,6	
Rib.1	(Sketch.1)	
Nº líneas	Nº puntos	Nº circunferencias
3	9	2
Nombre	Longitud (mm)	
Line.1	175	
Line.2	200	
Line.3	175	

Figura 4.32 Hoja 3 del análisis de la referencia 3.

La *Figura 4.31* y la *Figura 4.32* no están completas, ya que en la hoja aparecen los *sketches* de cada una de las operaciones que conforman ambas piezas.

Una vez observados los valores que CPIT extrae de ambas piezas, se puede pasar a examinar la información mostrada en la ventana de resultados del menú principal. Como se puede observar en la *Figura 4.24*, CPIT informa a través de la ventana que los módulos se están ejecutando correctamente. Como el número de operaciones de la pieza y de la referencia coincide, y existen fallos en los parámetros generales, la aplicación sigue ejecutándose para encontrar dónde se han cometido los mismos. Una vez compara las operaciones de ambas piezas, muestra una serie de consejos para que el usuario puede corregir su modelo.

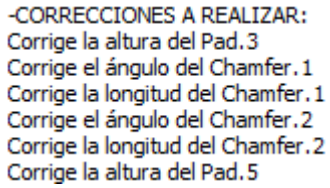


Figura 4.33 Consejos de CPIT para corregir la pieza 3.

En conclusión, CPIT ha analizado correctamente la pieza modelada por el alumno, dándole una calificación acorde al resultado obtenido. Además, los consejos mostrados a través de la ventana de resultados concuerdan con los fallos que tiene la pieza.

4.4 Discusión de resultados

El motivo por el que este proyecto se ha llevado a cabo es el desarrollo de una aplicación que permita autoevaluar piezas industriales 3D en contraste con piezas modelo o con plantillas modelo. Durante el transcurso del trabajo, se ha ido profundizando en la programación en VBA para lograr el objetivo marcado. El software obtenido ofrece la posibilidad de autoevaluar modelos de piezas 3D de distinta índole, mediante una interfaz que resulta fácil de manipular para el usuario.

Las piezas que el programa desarrollado evalúa son las creadas en el módulo *Part Design* de CATIA, que ofrece un amplio catálogo de operaciones con las que se pueden modelar una gran variedad de piezas. Como se vio en el *Capítulo 3*, concretamente en la sección *3.2.8 AnalisisOperaciones*, el software desarrollado es capaz de analizar dieciséis de estas operaciones, dando una clara libertad al usuario para diseñar su modelo.

Las operaciones de los menús *Transformation Features* y *Dress-Up Features* son analizadas correctamente por el programa desarrollado. Por otro lado, las operaciones del menú *Sketch-Based Features* son también, en la mayoría de los casos, correctamente analizadas. El programa desarrollado es capaz de identificar los parámetros que se nombraron en la sección 3.2.8, como la altura del *pad*, la profundidad del *pocket* o el tipo del *hole*, entre otros. Sin embargo, la operación *Multi-sections Solid*, que se basa en unir de una serie de *sketches* para formar un sólido, presenta una dificultad. A la hora de aplicar esta sobre la pieza modelada, algunos de los *sketches* que intervienen en la misma se quedan en el árbol de modelado, y no quedan confinados dentro de la operación. Consecuentemente, el programa desarrollado no es capaz de detectarlos, por lo que no son analizados. Esto se debe a los pasos que sigue el código empleado para detectar los *sketches* de una operación:

1. Seleccionar en el árbol de modelado la operación a analizar.
2. Contar los *sketches* que se encuentran confinados en la selección realizada.

Por tanto, si el *sketch* no está confinado en la operación, no es detectado.

Por último, para asegurar el correcto funcionamiento del software desarrollado, se realizó un período de prueba en el que se analizaron piezas de distinta índole. El objetivo era comprobar que los resultados obtenidos eran correctos y que las puntuaciones que el programa daba a las piezas eran acordes a los fallos encontrados durante los procesos de análisis. Se concluyó que el software desarrollado analizaba por lo general las piezas sin dificultad alguna y que la puntuación que establecía se adecuaba a los errores cometidos por el usuario. Además, se comprobó que el programa desarrollado cumplía en gran medidad los objetivos marcados al comienzo del proyecto, resultando ser una herramienta fácil de manipular e instructiva para los usuarios que la emplean, de ahí su posible aplicación en la docencia. A pesar de las ventajas que ofrece el software, también tiene una serie de aspectos a mejorar. Además del mencionado problema que aparece al analizar la operación *Multi-sections Solid*, cabe destacar que el tiempo de ejecución en algunos casos es grande. Para piezas con un número grande de operaciones, el tiempo de ejecución puede llegar incluso a un par de minutos, aunque comparándolo con el tiempo que se tardaría en corregir la pieza operación por operación, no resulta excesivo.

5 Conclusiones y líneas de trabajo futuro

Una vez finalizado el proyecto se puede valorar la línea de trabajo seguida y los resultados obtenidos, evaluando si los objetivos marcados al comienzo se han cumplido.

Partiendo de una idea clara, desarrollar una aplicación para autocorreción de ejercicios de modelado 3D, se ha conseguido obtener un software versátil capaz de autoevaluar piezas de distinta índole. La metodología empleada ha permitido obtener un programa que estima la puntuación de una pieza en base a las discrepancias encontradas con una referencia. Adicionalmente, la aplicación facilita al usuario una serie de consejos para mejorar la nota establecida, siendo una clara motivación para que el usuario comprenda los errores cometidos y trate de solventarlos.

Se han llevado a cabo multitud de pruebas del programa, obteniendo en su mayoría buenos resultados. Dentro del amplio catálogo de operaciones que ofrece el módulo *Part Design* de CATIA, CPIT es capaz de analizar dieciséis operaciones repartidas entre las ventanas *Sketch-Based Features*, *Transformation Feautures* y *Dress-Up Features*, ofreciendo una amplia libertad al usuario a la hora modelar su pieza. Además, se ha incluido la posibilidad de analizar modelos en los que se use *Geometrical Set* y modelos parametrizados con la herramienta *Formula*. El software desarrollado analiza parámetros generales de la pieza, como el volumen o los momentos principales de inercia, las operaciones que aparecen tanto en el árbol de modelado de la pieza del usuario como en el de la referencia y los *sketches* asociados a estas operaciones. Una vez estudiado el modelo, exporta los resultados a un archivo Excel de tres hojas, asociadas a los distintos niveles de análisis que realiza el programa.

Es importante mencionar que en algunos casos el rendimiento del programa no es óptimo. El tiempo de ejecución para piezas complejas, de un número alto de operaciones, es elevado. Esto se debe a la gran cantidad de líneas de código del programa y a la complejidad de ciertos comandos, lo que produce un retraso en la evaluación. Por ejemplo, para piezas como la *Pieza 3* el tiempo de ejecución es de un par de minutos aproximadamente, aunque comparándolo con el tiempo que se tardaría en analizar la pieza manualmente operación por operación, no resulta excesivo. Adicionalmente, hay operaciones que no se analizan correctamente en algunos casos, como el *Multi-sections Solid*. La complejidad de esta reside en que los *sketches* no aparecen siempre confinados en la operación, lo que provoca que CPIT no sea capaz de asociarlos a ella.

En conclusión, se ha desarrollado un software, que pretende solventar el problema que se presenta a la hora de corregir mecánicamente ejercicios de modelado de piezas industriales 3D, ofreciendo una posible vía para la autocorrección automatizada de los mismos. CPIT es un programa versátil, fácil

de usar e instructivo, dando una serie de consejos al usuario para mejorar su pieza y estableciendo una puntuación acorde a los errores encontrados durante el proceso de análisis. Por esta razón se piensa que tiene una clara aplicación en la docencia, ofreciendo a los alumnos una vía para preparar mejor sus exámenes al poder autoevaluar las piezas que realizan en clase, evitando que se tengan que parar operación por operación para ver si sus modelos se adecúan al del docente. Pese a todas las ventajas que ofrece la aplicación, cabe destacar que hay casos en los que su rendimiento no es el óptimo. Por esta razón, se aportan una serie de ideas y mejoras que podrían realizarse en el proyecto para completarlo.

- Extender el campo de aplicación del programa a otros módulos de CATIA. Esto permitiría tener una aplicación muy versátil, capaz de analizar cualquier modelo realizado en este software.

- Optimizar los códigos empleados para reducir el tiempo de ejecución del programa.

- Solucionar los problemas que da la operación *Multi-sections solid*. Se recomienda encontrar algún comando que permita identificar el nombre de los *sketches* asociados a la operación.

- Aumentar el número de operaciones analizadas. CATIA ofrece un amplio catálogo de operaciones que podrían ser en su mayoría implementadas en la aplicación.

Bibliografía

[1]. Laura García Ruesgas, Francisco Valderrama Gual, Cristina Torrecillas y Amparo Verdú. *Propuestas tecnológicas de autocorrección de ejercicios de modelado 3D*. *Revista ABE* (*Advances in Building Education*), 4(2), 42-59. *http://polired.upm.es/index.php/abe/article/view/4463*

[2]. Félix Sanz Adán y Julio Blanco Fernández. *CAD-CAM: gráficos, animación y simulación por computador*. Ed. Madrid: Thomson-Paraninfo, 2013.

[3]. José Ignacio Rojas Sola y Miguel Ángel Rubio Paramio. *Diseño Asistido por Ordenador*. Ed. Jaén : Universidad de Jaén, Servicio de Publicaciones e Intercambio Científico, 1997.

[4]. CATIA. (5 de abril de 2022). En *Wikipedia*. *https://es.wikipedia.org/w/index.php?title=CATIA&oldid=142733208*

[5]. Cristina Torrecillas. *Introducción a la programación Visual Basic en CATIA V5 y V6* (no publicado).

[6]. Emmett Ross. *VB Scripting for CATIA V5: how to program Catia macros*. Segunda Edición. Ed. Createspace, 2012.

[7]. Dassault Systèmes Web Site. IDL API Master Index. Disponible en: *http://catiadoc.free.fr/online/interfaces/CAAMasterIdx.htm* [Consulta: 30 de mayo de 2022].

Manual de usuario

1. Índice

2. Objetivo

Establecer los pasos específicos para ejecutar la aplicación correctamente, con el fin de que el usuario pueda comprobar la validez de su pieza.

3. Cómo descargar y ejecutar la macro

Los pasos a seguir son:

1. Descargar el archivo del programa, que tiene extensión *.catvba. Una vez descargado, se tiene que asegurar que la ruta de acceso al software no contenga caracteres especiales, como $, ni caracteres específicos del castellano como la letra *ñ*.

2. Abrir CATIA y clicar en la pestaña *Tools*. Dentro de ella, acceder a *Macro* y posteriormente clicar en *Macros*. También se puede llegar a la misma ventana pulsando *Alt* + *F8*.

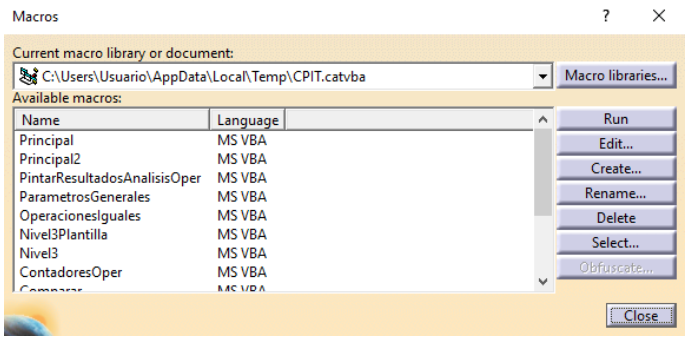


Figura A.1 Ventana de macros en CATIA.

3. Una vez se abra la ventana de la *Figura A.1*, debe clicar en *Macro libraries* y posteriormente en *Add existing library*. Al realizar este paso, se le abrirá una ventana en la que podrá escoger la macro del programa previamente descargado.

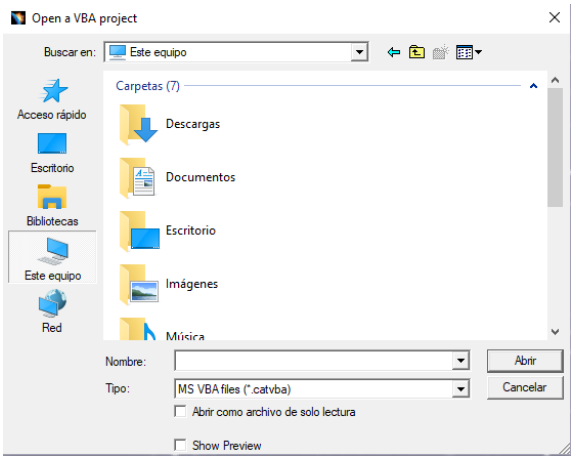


Figura A.2 Ventana para añadir una nueva macro.

4. Una vez añadida la macro, tiene que seleccionar el módulo denominado *Abrir_CPIT*, cuya función es ejecutar el software. Una vez seleccionado, simplemente debe clicar el botón denominado *Run* para abrir el menú del programa.

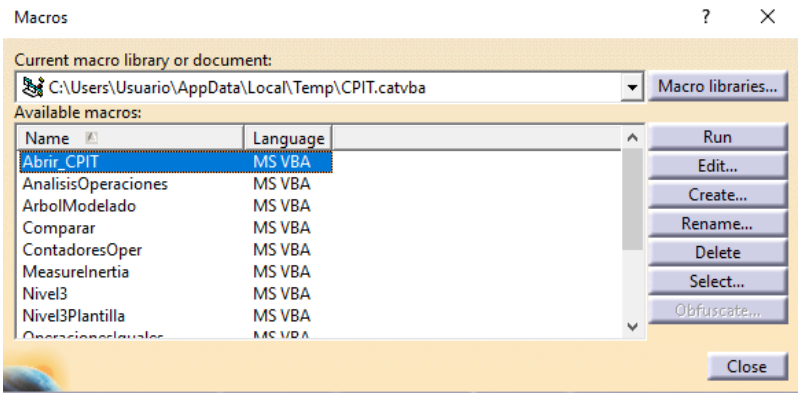


Figura A.3 Ejecutar macro.

4. Interacción con el menú principal

Una vez ejecutada la aplicación aparecerá el menú principal. Como se puede observar en la *Figura A.4*, se refleja en el cuadro superior del menú el directorio de la pieza que se desea analizar.

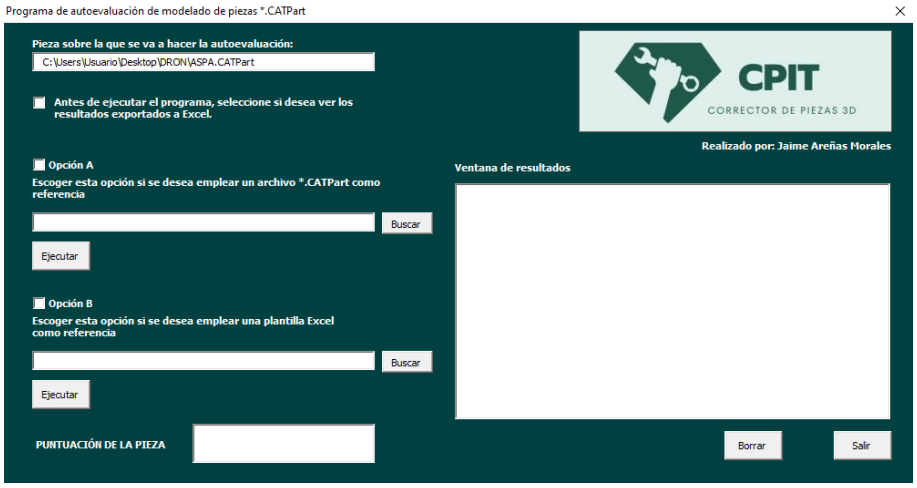


Figura A.4 Menú Principal de la aplicación.

El diseño del menú está pensado para que el usuario pueda interaccionar con el mismo, eligiendo distintas formas de analizar su pieza.

4.1 Mostrar resultados obtenidos

Una opción que ofrece el menú del programa es si se desean ver los resultados obtenidos tras el proceso de análisis. Si se activa la casilla, se mostrará el Excel con las soluciones una vez finalice la ejecución del programa. Independientemente de la elección tomada, el archivo Excel con las respuestas se guarda en la misma carpeta en la que esté almacenada la referencia escogida.

Figura A.5 Mostrar los resultados del programa.

4.2 Opciones de análisis que ofrece la aplicación

CPIT tiene la capacidad de comparar una pieza con otra de referencia o de compararla con una plantilla Excel en la que se recojan los parámetros adecuados, opción A y opción B respectivamente. Cuando se seleccione una de las dos opciones, el software aplica la herramienta *Measure Inertia* sobre la pieza modelada por el usuario, que es fundamental para extraer los parámetros generales del modelo. Si se vuelve a marcar la misma opción, se actualizan las medidas inerciales por si se ha realizado alguna modificación en la pieza. Además, al escoger una de las opciones la otra queda desactivada, evitando que el usuario pueda ejecutarla.

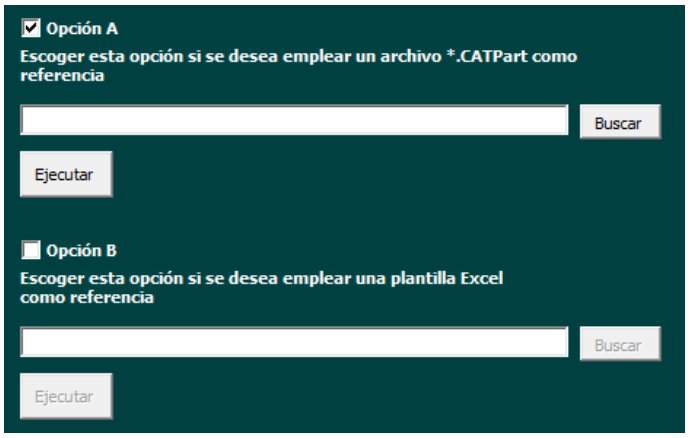


Figura A.6 Elección de la opción de análisis.

Una vez marcada la opción, se tiene que buscar el archivo de referencia. Para ello, se tiene que clicar sobre el botón *Buscar*, abriéndose una ventana para realizar la selección del archivo.

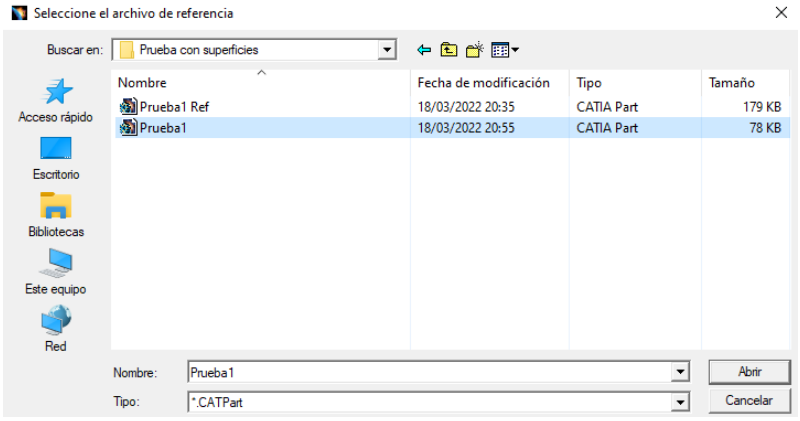


Figura A.7 Ventana para seleccionar el directorio del archivo de referencia.

Una vez abierto el archivo, queda su directorio guardado en el cuadro de texto adyacente. Adicionalmente, si la opción marcada es la A, se abre la pieza de referencia en CATIA y se le aplica la herramienta *Measure Inertia*, al igual que a la pieza modelada por el usuario.

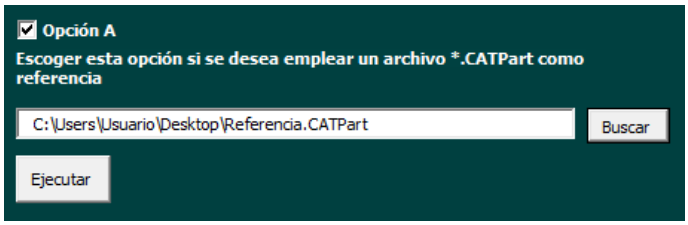


Figura A.8 Selección del archivo de referencia.

Por último, se debe clicar en el botón *Ejecutar* para que comience el proceso de análisis. Una vez se obtengan los resultados, aparecerá la puntuación que el software asocia al modelo del usuario, en base a los errores encontrados durante el análisis.

Figura A.9 Puntuación de la pieza estimada por el programa.

4.3 Interpretación de la ventana de resultados

Mientras el programa se está ejecutando, en la ventana de resultados va apareciendo si los distintos niveles de análisis se están llevando a cabo correctamente. Además, aparece información adicional relacionada con cada nivel:

- Nivel 1: Aparece *CORRECTO* si los parámetros generales de la pieza coinciden con los de la referencia e *INCORRECTO* en caso contrario.
- Nivel 2: Aparece *CORRECTO* si el número de operaciones de la pieza coincide con el de la referencia e *INCORRECTO* en caso contrario. Además, se muestran las correcciones que el usuario debe realizar. Estas están relacionadas con los parámetros tridimensionales de las operaciones (no con el sketch).
- Nivel 3: Solo se indica si se han analizado los *sketches* correctamente.

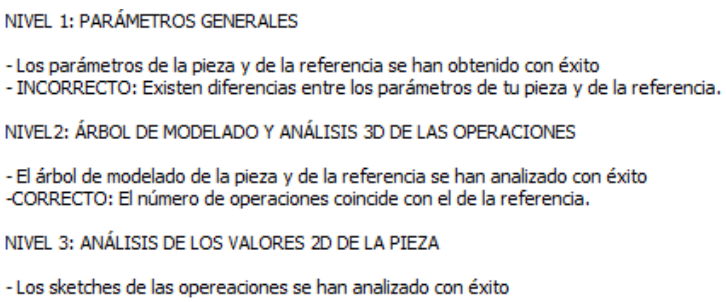


Figura A.10 Ejemplo de la información que muestra la ventana de resultados.

En caso de que el usuario no quiera ver los resultados a Excel, esta ventana será de ayuda para saber si el programa se está ejecutando correctamente y si su pieza tiene algún fallo.

4.4 Borrar menú y salir del programa

En la parte inferior del menú principal hay dos botones que permiten borrar los datos del menú previamente introducidos o salir de la aplicación.

Figura A.11 Borrar datos del menú y salir de la aplicación.

4.5 Errores que el usuario debe evitar

CPIT avisa al usuario en caso de que cometa un error previo a ejecutar el programa. Por ejemplo, si no introduce el directorio del archivo de referencia, aparece un mensaje que informa del error cometido.

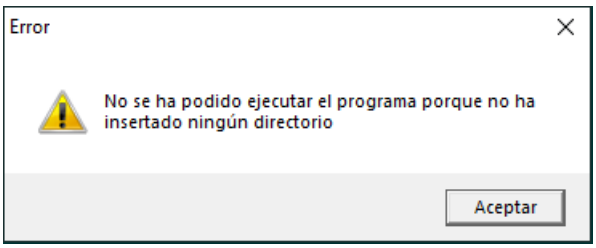


Figura A.12 Error por no introducir el directorio del archivo de referencia.

También pude ocurrir que el usuario abra la macro sin abrir previamente la pieza que desea analizar. En ese caso, la aplicación avisa de que no es posible realizar el análisis al no haber una pieza abierta en CATIA.

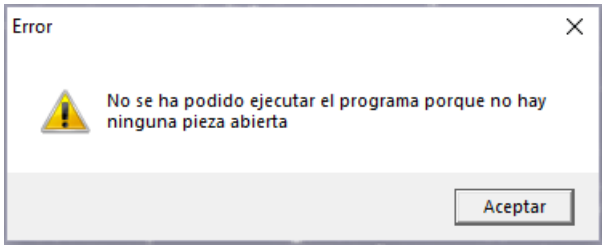


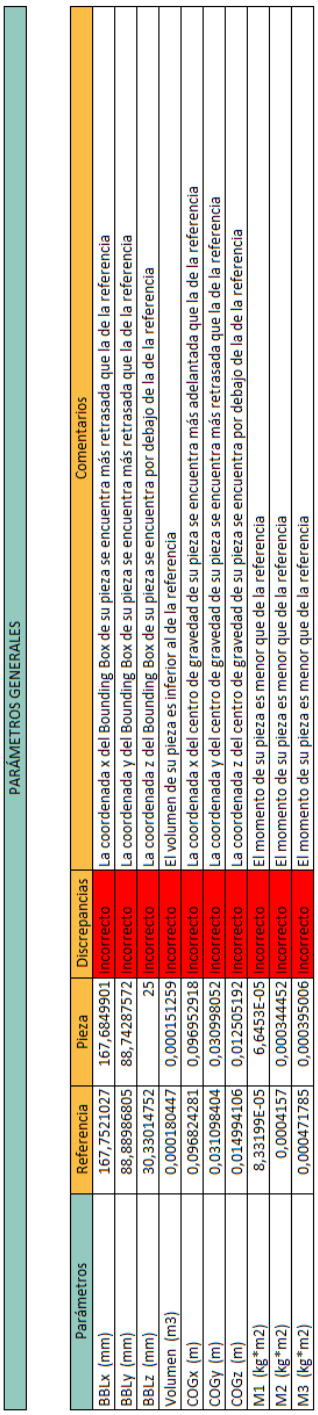
Figura A.13 Error por no abrir una pieza antes de ejecutar la macro.

5. Interpretación del archivo Excel de resultados

Si el usuario selecciona que sí desea ver los resultados, la aplicación tiene dos formas de actuar:

- En caso de tener una pieza de referencia, se genera un archivo Excel de tres hojas en las aparecerán los resultados de cada nivel.
- En caso de tener una plantilla Excel de referencia, la aplicación lee la información almacenada en la plantilla, y genera y completa un archivo Excel de tres hojas en las aparecerán los resultados de cada nivel.

En la primera hoja aparecen los parámetros generales de la pieza y de la referencia (Nivel 1). Estos se muestran en una tabla en la que se indica el nombre del parámetro, sus unidades, los valores que toma en la pieza y en la referencia, las discrepancias entre los mismos y comentarios que ayudan al usuario a corregir los errores existentes.

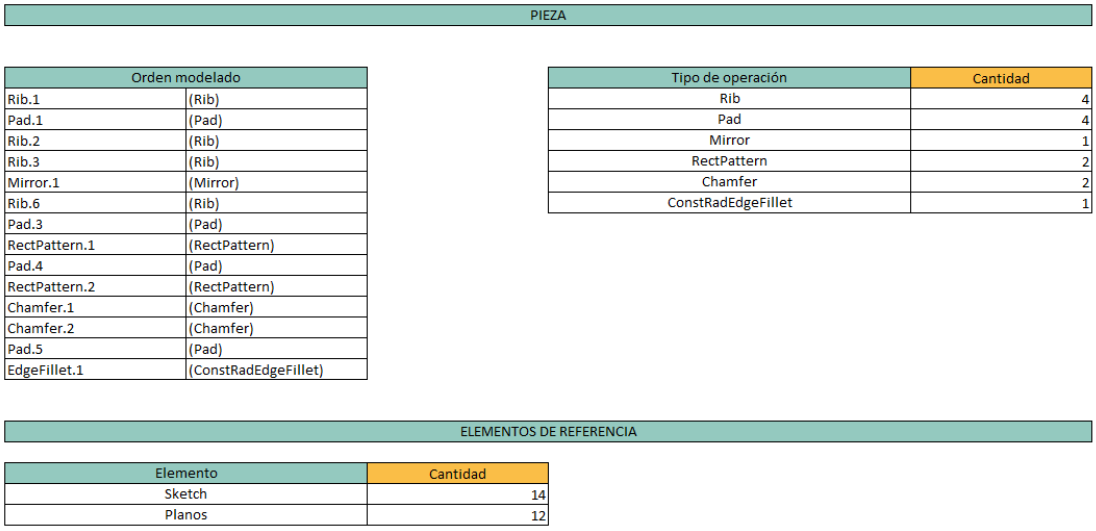


PARÁMETROS GENERALES

Parámetros	Referencia	Pieza	Discrepancias	Comentarios
BBLx (mm)	167,7521027	167,6849901	Incorrecto	La coordenada x del Bounding Box de su pieza se encuentra más retrasada que la de la referencia
BBLy (mm)	88,88986805	88,74287572	Incorrecto	La coordenada y del Bounding Box de su pieza se encuentra más retrasada que la de la referencia
BBLz (mm)	30,33014752	25	Incorrecto	La coordenada z del Bounding Box de su pieza se encuentra por debajo de la de la referencia
Volumen (m3)	0,000180447	0,000151259	Incorrecto	El volumen de su pieza es inferior al de la referencia
COGx (m)	0,096824281	0,096952918	Incorrecto	La coordenada x del centro de gravedad de su pieza se encuentra más adelantada que la de la referencia
COGy (m)	0,031098404	0,030998052	Incorrecto	La coordenada y del centro de gravedad de su pieza se encuentra más retrasada que la de la referencia
COGz (m)	0,014994106	0,012505192	Incorrecto	La coordenada z del centro de gravedad de su pieza se encuentra por debajo de la de la referencia
M1 (kg*m2)	8,33199E-05	6,6453E-05	Incorrecto	El momento de su pieza es menor que de la referencia
M2 (kg*m2)	0,0004157	0,000344452	Incorrecto	El momento de su pieza es menor que de la referencia
M3 (kg*m2)	0,000471785	0,000395006	Incorrecto	El momento de su pieza es menor que de la referencia

Figura A.14 Hoja 1 de resultados.

En la segunda hoja aparece el árbol de modelado y el número de operaciones de la pieza y de la referencia. Además,se muestran los parámetros tridimensionales de aquellas operaciones que pertenezcan tanto al árbol de modelado de la pieza como al de la referencia. Es decir, esta hoja recoge el Nivel 2 de la aplicación.



PIEZA

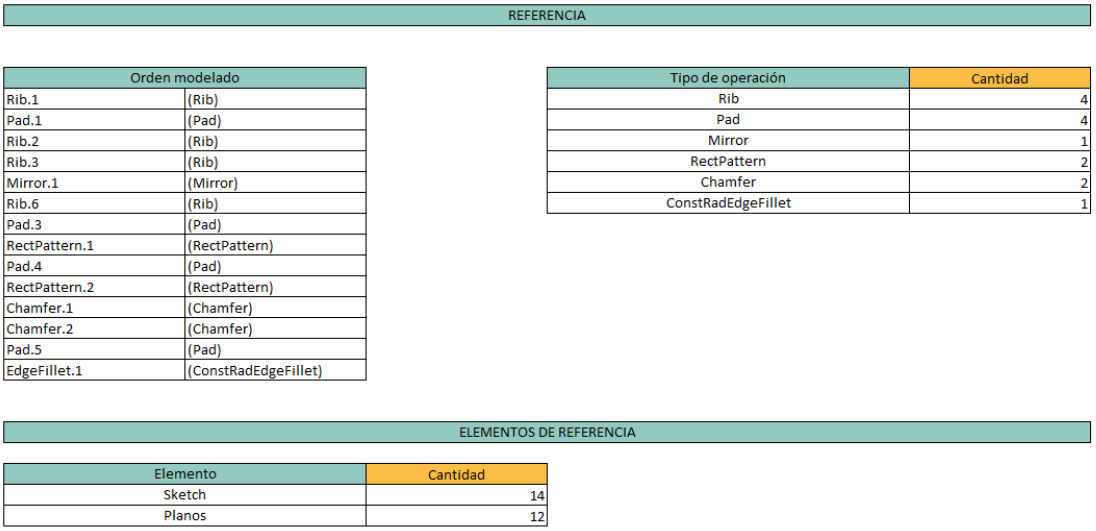
Orden modelado	
Rib.1	(Rib)
Pad.1	(Pad)
Rib.2	(Rib)
Rib.3	(Rib)
Mirror.1	(Mirror)
Rib.6	(Rib)
Pad.3	(Pad)
RectPattern.1	(RectPattern)
Pad.4	(Pad)
RectPattern.2	(RectPattern)
Chamfer.1	(Chamfer)
Chamfer.2	(Chamfer)
Pad.5	(Pad)
EdgeFillet.1	(ConstRadEdgeFillet)

Tipo de operación	Cantidad
Rib	4
Pad	4
Mirror	1
RectPattern	2
Chamfer	2
ConstRadEdgeFillet	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	14
Planos	12

Figura A.15 Hoja 2. Parte 1 (Pieza).



REFERENCIA

Orden modelado	
Rib.1	(Rib)
Pad.1	(Pad)
Rib.2	(Rib)
Rib.3	(Rib)
Mirror.1	(Mirror)
Rib.6	(Rib)
Pad.3	(Pad)
RectPattern.1	(RectPattern)
Pad.4	(Pad)
RectPattern.2	(RectPattern)
Chamfer.1	(Chamfer)
Chamfer.2	(Chamfer)
Pad.5	(Pad)
EdgeFillet.1	(ConstRadEdgeFillet)

Tipo de operación	Cantidad
Rib	4
Pad	4
Mirror	1
RectPattern	2
Chamfer	2
ConstRadEdgeFillet	1

ELEMENTOS DE REFERENCIA

Elemento	Cantidad
Sketch	14
Planos	12

Figura A.16 Hoja 2. Parte 1 (Referencia).

Debido a que el ancho de las columnas es muy elevado para que todos los datos queden confinados en sus respectivas tablas, se ha optado por poner los resultados obtenidos para la pieza y para la referencia por separado. (*Figura A.15* y *Figura A.16* respectivamente).

A continuación se muestran los resultados obtenidos del análisis tridimensional de las operaciones de la pieza (*Figura A.17*) y de la referencia (*Figura A.18*).

PIEZA

Pad

Nombre	Altura (mm)
Pad.1	25

ConstRadEdgeFillet

Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)
EdgeFillet.1	5	1	25

Hole

Nombre	Tipo	Diámetro (mm)	Profundidad (mm)	Extensión	Diámetro Superior (mm)	Profundidad Superior (m	Ángulo (º)
Hole.1	0	10	25	2	-	-	-
Hole.3	0	10	20	2	-	-	-

Pocket

Nombre	Profundidad (mm)
Pocket.1	25

Figura A.17 Hoja 2. Parte 2 (Pieza).

REFERENCIA

Pad

Nombre	Altura (mm)
Pad.1	30

Pocket

Nombre	Profundidad (mm)
Pocket.1	30

Hole

Nombre	Tipo	Diámetro (mm)	Profundidad (mm)	Extensión	Diámetro Superior (mm)	Profundidad Superior (m	Ángulo (º)
Hole.1	4	10	30	2	15	5	60
Hole.2	4	10	20	2	15	5	60

ConstRadEdgeFillet

Nombre	Radio (mm)	Nº redondeos	Longitud 1 (mm)
EdgeFillet.1	5	1	30

Figura A.18 Hoja 2. Parte 2 (Referencia).

Por último, en la tercera hoja aparece el análisis de los sketches de cada operación (Nivel 3). En este nivel solo se analizan los valores 2D de aquellas operaciones que pertenezcan tanto a la pieza como a la referencia.

PIEZA

Pad.1	(Sketch.1)

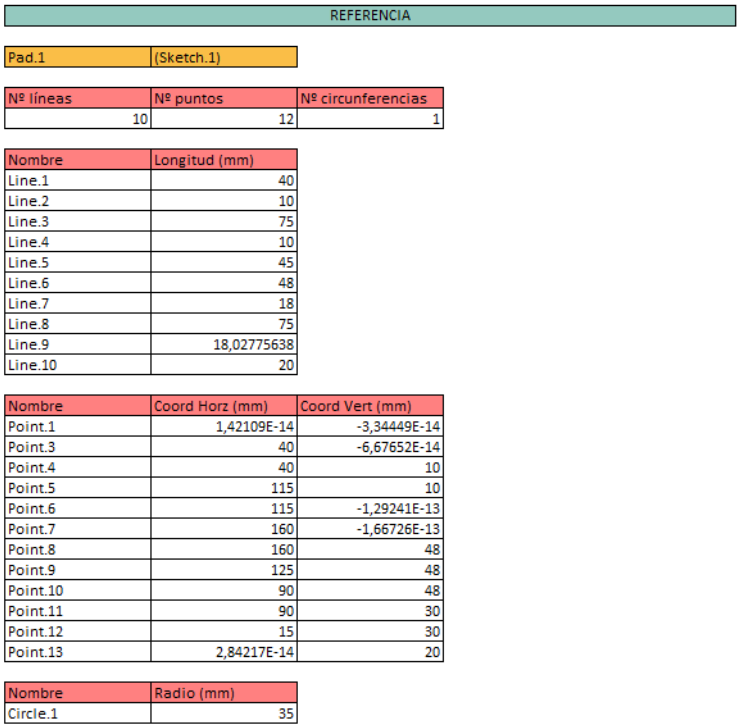
Nº líneas	Nº puntos	Nº circunferencias
10	12	1

Nombre	Longitud (mm)
Line.1	40
Line.2	10
Line.3	75
Line.4	10
Line.5	45
Line.6	48
Line.7	18
Line.8	75
Line.9	18,02775638
Line.10	20

Nombre	Coord Horz (mm)	Coord Vert (mm)
Point.2	40	0
Point.3	40	10
Point.4	115	10
Point.5	115	0
Point.6	160	0
Point.7	160	48
Point.8	125	48
Point.9	90	48
Point.10	90	30
Point.11	15	30
Point.12	0	20
Point.13	0	0

Nombre	Radio (mm)
Circle.1	35

Figura A.19 Hoja 3 (Pieza).



REFERENCIA

Pad.1	(Sketch.1)

Nº líneas	Nº puntos	Nº circunferencias
10	12	1

Nombre	Longitud (mm)
Line.1	40
Line.2	10
Line.3	75
Line.4	10
Line.5	45
Line.6	48
Line.7	18
Line.8	75
Line.9	18,02775638
Line.10	20

Nombre	Coord Horz (mm)	Coord Vert (mm)
Point.1	1,42109E-14	-3,34449E-14
Point.3	40	-6,67652E-14
Point.4	40	10
Point.5	115	10
Point.6	115	-1,29241E-13
Point.7	160	-1,66726E-13
Point.8	160	48
Point.9	125	48
Point.10	90	48
Point.11	90	30
Point.12	15	30
Point.13	2,84217E-14	20

Nombre	Radio (mm)
Circle.1	35

Figura A.20 Hoja 3 (Referencia).

6. FAQs

A continuación se recogen algunas preguntas que le pueden surgir al usuario al interactuar con la aplicación.

1. ¿Es posible analizar una pieza con más de un Body?

En principio no. CPIT es un proyecto que se ha desarrollado en base a la idea de comparar una pieza con una referencia, determinando las discrepancias existentes entre ambos modelos y estableciendo una puntuación acorde a los errores encontrados.

2. ¿La aplicación modifica los parámetros de la pieza?

No. Lo que se fomenta es que el usuario tenga constancia de los errores cometidos y pueda cambiarlos. Si CPIT los corrigiera automáticamente, el aspecto docente se perdería por completo.

3. ¿Se pueden analizar objetos realizados con otros módulos de CATIA V5R19?

No. En principio CPIT está pensado para la corrección de piezas realizadas en la herramienta *Part Design* de CATIA V5R19.